DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA 93943-5002

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

APPLICATIONS OF EIGENSTRUCTURE ASSIGNMENT TO DESIGN OF ROBUST MIMO DECOUPLING CONTROLLERS AND TO RECONFIGURATION ALGORITHMS FOR DAMAGED FLIGHT CONTROL SYSTEMS

by

Valentin F. Gavito, Jr.

December 1986

Thesis Advisor D.J. Collins

Approved for public release; distribution is unlimited.

T230488

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
2b DECLASSIFICATION / DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)

6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School	6b OFFICE SYMBOL *(If applicable)* 31	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c ADDRESS *(City, State, and ZIP Code)* Monterey, California 93943 - 5000		7b ADDRESS *(City, State, and ZIP Code)* Monterey, California 93943 - 5000
8a NAME OF FUNDING / SPONSORING ORGANIZATION	8b OFFICE SYMBOL *(If applicable)*	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c ADDRESS *(City, State, and ZIP Code)*

10 SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO

11 TITLE *(Include Security Classification)* APPLICATIONS OF EIGENSTRUCTURE ASSIGNMENT TO DESIGN OF ROBUST MIMO DECOUPLING CONTROLLERS AND TO RECONFIGURATION ALGORITHMS FOR DAMAGED FLIGHT CONTROL SYSTEMS

12 PERSONAL AUTHOR(S)
Gavito, Valentin F., Jr.

13a TYPE OF REPORT Ph.D. Thesis	13b TIME COVERED FROM ______ TO ______	14 DATE OF REPORT *(Year, Month, Day)* 1986 December	15 PAGE COUNT 232

16 SUPPLEMENTARY NOTATION

17 COSATI CODES

FIELD	GROUP	SUB-GROUP

18 SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Multivariable Control, Robust Controllers, Eigenstructure Assignment, Reconfiguring Controllers

19 ABSTRACT *(Continue on reverse if necessary and identify by block number)*

A matrix treatment of eigenstructure assignment theory as applied to the generalized linear time invariant system is presented. New geometrical interpretations of the constraints on selecting desired closed loop right and left eigenvectors as functions of the open loop system parameters are then shown to give qualitative measures on the computational complexity of selecting the vectors. Numerical optimization schemes are then presented which satisfy these geometric interpretations for subsequent encoding of an interactive eigenstructure synthesis program, EIGENS. Robust decoupling controllers are then synthesized via EIGENS. A new theoretical application of eigenstructure assignment to reconfiguring damaged digital flight controllers is then presented. The thesis concludes with reconfigured solution demonstrations as applied to the F/A-18 aircraft.

20 DISTRIBUTION / AVAILABILITY OF ABSTRACT ☐ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED

22a NAME OF RESPONSIBLE INDIVIDUAL D.J. Collins	22b TELEPHONE *(Include Area Code)* 408-646-2826	22c OFFICE SYMBOL 67Co

DD FORM 1473, 84 MAR

83 APR edition may be used until exhausted
All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

Approved for public release: distribution is unlimited

Applications of Eigenstructure Assignment to Design of Robust MIMO Decoupling Controllers and to Reconfiguration Algorithms for Damaged Flight Control Systems

by

Valentin F. Gavito, Jr.
Commander, United States Navy
B.S.M.E., Southern Methodist University, 1970
M.S.A.E., The Naval Postgraduate School, 1976

Submitted in partial fulfillment of the
requirements for the degree

DOCTOR OF PHILOSOPHY IN AERONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1986

ABSTRACT

A matrix treatment of eigenstructure assignment theory as applied to the generalized linear time invariant system is presented. New geometrical interpretations of the constraints on selecting desired closed loop right and left eigenvectors as functions of the open loop system parameters are then shown to give qualitative measures on the computational complexity of selecting the vectors. Numerical optimization schemes are then presented which satisfy these geometric interpretations for subsequent encoding of an interactive eigenstructure synthesis program, EIGENS. Robust decoupling controllers are then synthesized via EIGENS. A new theoretical application of eigenstructure assignment to reconfiguring damaged digital flight controllers is then presented. The thesis concludes with reconfigured solution demonstrations as applied to the F/A-18 aircraft.

TABLE OF CONTENTS

I. INTRODUCTION 11

II. BACKGROUND 13

III. EIGENSTRUCTURE THEORY FOR LINEAR TIME INVARIANT SYSTEMS 17

A. PRELIMINARIES 17

1. The Singular Value Decomposition 17

2. The Matrix Pseudo Inverse 19

B. THE GENERALIZED FEEDBACK EXPRESSION FOR LINEAR TIME INVARIANT SYSTEMS 21

C. RIGHT AND LEFT EIGENSPACE: A GEOMETRIC INTERPRETATION 30

IV. NUMERICAL OPTIMIZATION 44

A. THE AUTOMATIC DESIGN SYNTHESIS (ADS) PROGRAM 44

B. THE NUMERICAL OPTIMIZATION PROBLEM 47

1. State Feedback Controller Synthesis 48

2. Output Feedback Controller Synthesis 49

3. A Robust State Feedback Controller Synthesis 51

4. Feedback Controller Synthesis via Eigenvalue Shifting 51

V. EIGENS: AN INTERACTIVE DESIGN ALGORITHM 53

A. DESIGN PHILOSOPHY 53

B. PROGRAM DESCRIPTION 54

C. SUBROUTINE KVECT 55

D. SUBROUTINE FEEDEF 56

VI. MIMO CONTROLLER DESIGN DEMONSTRATIONS 60

A. A ROBUST DECOUPLING STATE FEEDBACK CONTROLLER 60

B. A QUASI-OPTIMAL STATE FEEDBACK DECOUPLING CONTROLLER 64

C. A ROBUST OUTPUT FEEDBACK CONTROLLER 64

VII. EXPLOITATION OF EIGENSTRUCTURE ASSIGNMENT TO SELF RECONFIGURING AIRCRAFT MIMO CONTROLLERS: A NEW APPROACH 75

A. PROBLEM STATEMENT 75

B. ALGEBRAIC SOLUTION 76

C. RECONFIGURATION ALGORITHM 78

VIII. A RECONFIGURED SOLUTION DEMONSTRATION 80

A. BACKGROUND 80

B. F/A-18A SYSTEM OVERVIEW 80

C. F/A-18A LINEARIZED DYNAMIC MODEL 81

D. F/A-18A AUGMENTED LINEARIZED DYNAMIC MODEL 82

E. THE SYMMETRIC DEGRADATION PROBLEM 86

1. Case A 93

2. Case B 102

F. THE ASYMMETRIC DEGRADATION PROBLEM 109

IX. CONCLUSIONS AND RECOMMENDATIONS 131

APPENDIX A: NOTATION 132

APPENDIX B: CONSTRAINTS ON EIGENSTRUCTURE SPECIFICATION 134

APPENDIX C: EIGENS 137

APPENDIX D: RECONF 184

APPENDIX E: ERSPACE 196

APPENDIX F: ELSPACE 202

APPENDIX G: SAMPLE RECONFIGURATION RUN FOR F-18, CASE A DAMAGE 217

LIST OF REFERENCES 229

INITIAL DISTRIBUTION LIST 232

LIST OF TABLES

1. FEEDBACK EXPRESSIONS AND EIGENVECTOR CONSTRAINTS 28
2. CH-47 LATERAL DYNAMICS MODEL 61
3. CH-47 DECOUPLING STATE FEEDBACK DESIGNS 63
4. CH-47 OPTIMAL STATE FEEDBACK DESIGNS 65
5. L-1011 LATERAL DYNAMICS MODEL 66
6. L-1011 OUTPUT FEEDBACK DESIGNS 68
7. F/A-18A SYSTEM MATRICES FOR M = 0.6, ALT = 10,000 FT. 83
8. F/A-18A UNDAMAGED C-LOOP EIGENVALUES:M = 0.6 ALT = 10K 94
9. F/A-18A UNDAMAGED AIRCRAFT EIGENSTRUCTURE 95
10. F/A-18A UNDAMAGED FEEDBACK GAIN MATRIX 96
11. F/A-18A NOMINAL NULL SPACE RESIDUALS 97
12. CASE A (25% SYMMETRIC) DAMAGED C-LOOP EIGENVALUES 101
13. CASE A (25%SYMMETRIC DAMAGE) AIRCRAFT EIGENSTRUCTURE 102
14. CASE A NULL SPACE RESIDUALS 103
15. RECONFIGURED FEEDBACK GAINS FOR CASE A DAMAGE 104
16. CASE A RECONFIGURED AIRCRAFT EIGENSTRUCTURE 105
17. CASE B(50% SYMMETRIC) DAMAGED C-LOOP EIGENVALUES 113
18. CASE B (50% SYMMETRIC DAMAGE) AIRCRAFT EIGENSTRUCTURE 114
19. CASE B NULL SPACE RESIDUALS 115
20. RECONFIGURED FEEDBACK GAINS FOR CASE B DAMAGE 116
21. CASE B RECONFIGURED AIRCRAFT EIGENSTRUCTURE 117
22. PERFORMANCE RECOVERY FEEDBACK GAINS FOR CASE B DAMAGE 120
23. CASE B PERFORMANCE RECOVERY AIRCRAFT EIGENSTRUCTURE 121
24. CASE C(25% ASYMMETRIC) DAMAGED C-LOOP EIGENVALUES 124

25. CASE C (25% ASYMMETRIC DAMAGE) AIRCRAFT EIGENSTRUCTURE 125
26. RECONFIGURED FEEDBACK GAINS FOR CASE C DAMAGE 130

LIST OF FIGURES

3.1 Basic Lattice Diagram 36
3.2 Right State Space Lattice Diagram 37
3.3 Left State Space Lattice Diagram 38
3.4 Total Space Lattice Diagram 39
3.5 Case I: Full State Feedback ($m = n$) Lattice Diagram 40
3.6 Case II: Full State Feedback ($m < n$) Lattice Diagram 41
3.7 Case IV: Output Feedback ($m = n, l < n$) Lattice Diagram 42
3.8 Case V: Output Feedback ($m < n, l < n, m = l$) Lattice Diagram 43
5.1 EIGENS General Flowchart 57
5.2 Subroutine KVECT 58
5.3 Subroutine FEEDEF 59
6.1 $\sigma_{min}(I + FG)$ for CH-47 Decoupled Designs 69
6.2 $\sigma_{min}(I + FG)$ for CH-47 Optimal Designs 70
6.3 CH-47 Lateral Response for Optimal Designs 71
6.4 L-1011 Response to $\beta(0) = 1.0$ deg:Sobel 72
6.5 L-1011 Response to $\beta(0) = 1.0$ deg:EIGENS 73
6.6 $\sigma_{min}(I + FG)$ for L-1011 Decoupled Designs 74
8.1 F/A-18A Flight Control System 87
8.2 F/A-18A Flight Control Surfaces 88
8.3 F/A-18A Longitudinal Control Model 89
8.4 F/A-18A Lateral-Directional Control Model 90
8.5 F/A-18A Undamaged Longitudinal Response 92
8.6 Case A Damaged Pitch Attitude Response 98
8.7 Case A Damaged Pitch Rate Response 99
8.8 Case A Reconfigured Pitch Attitude Response 106
8.9 Case A Reconfigured Pitch Rate Response 107
8.10 Case B Damaged Pitch Attitude Response 110
8.11 Case B Damaged Pitch Rate Response 111

8.12 Case B Reconfigured Pitch Attitude Response 118
8.13 Case B Reconfigured Pitch Rate Response 119
8.14 Case B Performance Recovery Pitch Attitude Response 122
8.15 Case B Performance Recovery Pitch Rate Response 123
8.16 Case C Bank Angle Responses 126
8.17 Case C Yaw Rate Responses 127
8.18 Case C Pitch Attitude Responses 128
8.19 Case C Pitch Rate Responses 129

ACKNOWLEDGEMENTS

To the United States Navy, for which it has been an honor and privilege to serve for the past sixteen years, I express my sincere gratitude for being granted the opportunity to pursue all of my postgraduate education.

A special word of appreciation goes to Professor Dan Collins who has so patiently guided me through all facets of my post graduate research at the Naval Postgraduate School. During both my master and doctorate levels of study, he has been instrumental in ensuring that my *head remained out of the cockpit* when frustrated by the toils of academic research.

Also a word of thanks to Professor Dick Franke who provided guidance and helpful comments with regard to the matrix theory presented in this thesis and to the other members of my committee: Professors Max Platzer, Allen Fuhs, and Hal Titus who were always available to listen and provide positive strokes.

I must also acknowledge Professor Doug Williams and his staff at the W.R. Church Computer Center for their support and positive response to my requests for assistance.

Special words of appreciation go to my parents, Val and Nina Gavito who for so many years have provided the right ingredients of parental love, guidance, and support which gave me a solid foundation to pursue my academic and professional goals.

The words of love and gratitude truly go to my family. To Jennifer, Cory, and Chris who through the years have endured the hardships of leaving friends and adjusting to new surroundings. Final words of love and appreciation are reserved for my loving wife, Marcia, who has seen the best and the worst of me through all my academic and professional challenges.

I. INTRODUCTION

Modification of linear or non linear dynamic system behaviours via feedback control of either the estimated or observed variable or the dynamic state variable is a fundamental tool of modern control engineering. Such desired modifications might include [Ref. 1]:

- *a.* *Stabilizing otherwise unstable systems,*
- *b.* *Decoupling overall system response to plant or controller noise,*
- *c.* *Desensitizing system performance as a function of plant variations (robustness) or,*
- *d.* *Invoking a desired transient response characteristic and modal signature.*

For linear time invariant single input single output (*SISO*) systems, calculation of the required feedback to perform one of these desired system behaviours is relatively straight forward and well documented [Refs. 2,3]. The *SISO* static feedback problem is additionally characterized by a specific design specification. The classic pole placement problem for example results in solving a full rank linear system of equations for the unknown feedback gains. These feedback gains will then invoke a set of closed loop poles which in turn give rise to specific transient behaviours. Further modification of the modal behaviour of the system can only be accomplished through a different set of feedback gains. This limited degree of freedom of the feedback design space of the classical *SISO* dynamic problem can additionally be interpreted as a bound on what performance specifications one can designate. Specification of additional system modification such as modal decoupling for example is not possible.

Design of acceptable feedback compensations for linear time invariant multi input multi output (*MIMO*) systems are primarily treated by state space solution techniques of modern control. As modern control theory can also treat non-linear and time varying dynamic systems, this time domain approach has proved successful both computationally and analytically. The linear quadratic gaussian (*LQG*) problem is a prime example of a successful time domain solution approach [Refs. 4,5,6]. Frequency response methods, along with such eigenvalue design approaches as characteristic loci and dyadic expansions and non - eigenvalue methods as the inverse Nyquist array technique have also been noted in the literature [Ref. 7]. More importantly, the underlying theme in all these design methodologies is that the resulting set of feedback gains, *F*, which satisfy a singular design criteria such as eigenvalue placement, maximizing a multivariable robustness measure [Ref. 8] or minimizing a particular quadratic cost function [Ref. 9] for example, is not unique. This non uniqueness of solution ideally allows the multivariable controller design problem to be formulated for solution by numerical optimization techniques.

Exploiting the specification of the eigenvectors of a *MIMO* closed loop system via numerical optimization techniques is the prime motivation of this research. Results include the following.

a. An interactive algorithm for eigenstructure synthesis and analysis of MIMO systems.

b. An analysis of both left and right eigenspaces as ***geometric*** interpretations of algebraic restrictions on closed loop eigenvector specification published previously, and

c. A *new* application of eigenstructure assignment to reconfiguring digital flight controllers for a class of damaged flight control systems.

Subsequent to a discussion of previous reported research in eigenstructure assignment, a new matrix treatment of eigenstructure theory is presented. This is followed by a description of the numerical optimization techniques used for designing particular multivariable controllers. Applications of these numerical techniques to design of various robust decoupling controllers are then shown. The latter part will present the theoretical framework of a new application of eigenstructure assignment to the synthesis of reconfigured digital flight controllers. The thesis will then conclude with reconfigured solutions for specific classes of damage to the F-18 tactical aircraft.

II. BACKGROUND

The following state variable equations define the linear time invariant system used throughout this thesis for the continuous time domain. The state, output, and control equations are,

$$\dot{x} = Ax + Bu + G_1\delta \qquad \text{(eqn 2.1)}$$

$$y = Cx + Du \qquad \text{(eqn 2.2)}$$

$$u = Fy + G_2\delta \qquad \text{(eqn 2.3)}$$

where each vector is defined as elements of the following vector spaces,

$$x \in \mathscr{R}^n \qquad \text{(eqn 2.4)}$$

$$y \in \mathscr{R}^l \qquad \text{(eqn 2.5)}$$

$$u \in \mathscr{R}^m \qquad \text{(eqn 2.6)}$$

$$\delta \in \mathscr{R}^c \qquad \text{(eqn 2.7)}$$

The individual system matrices are defined as follows,

A = n x n plant matrix

B = n x m control matrix

G_1 = n x c feed forward command state matrix

C = l x n output matrix

D = l x m feed forward output matrix

F = m x l feedback gain matrix

G_2 = m x c feed forward command input matrix

The flexibility afforded by eigenstructure assignment to design of linear time invariant *MIMO* state feedback controllers has been well documented since the mid 1970's. Moore [Ref. 10] defined this flexibility of design beyond specification of closed loop eigenvalues in terms of ***allowable*** sets of closed loop eigenvectors for systems with distinct eigenvalues. In addition, he demonstrated that for those eigenvalues which were invariant under state feedback (*uncontrollable*), design freedom of eigenvector specification still existed without rigorous constraints. During the same research period, Srinathkumar [Ref. 11] showed that for controllable systems with ***n states and m inputs, n closed loop eigenvalues and 'm x n' elements*** of the corresponding eigenvector matrix may be arbitrarily specified. A key additional constraint is that no more then ***'m' elements*** of each individual eigenvector may be specified simultaneously. Klein and Moore [Ref. 12] extended the results of Moore [Ref. 10] to include systems with a given set of non-distinct eigenvalues. The algorithm presented by these researchers enables a designer to invoke an ***allowable*** closed loop Jordan eigenstructure via state feedback.

For *MIMO* systems employing ***output*** feedback, Srinathkumar [Ref. 13] presented an eigenstructure design theorem that has become a foundation for many researchers engaged in analysis of such *MIMO* systems. Restated here,

> *for systems with 'm' inputs and 'l' outputs, max (m,l) closed loop eigenvalues can be specified and max (m,l) eigenvectors or reciprocal vectors by duality can be partially assigned with min (m,l) elements of each eigenvector specified.*

As the analytical treatment of eigenstructure assignment became familiar, subsequent workers began to explore applications to some existing design problems. Two examples are noted. Sebakhy and Abdel - Moneium [Ref. 14] presented an algorithm for computing state feedback gains which invoked minimum time responses (deadbeat controllers) for multivariable linear discrete time systems which simultaneously allowed the designer flexibility in selecting the closed loop eigenvectors. Klein [Ref. 15] provided guidelines for constructing state feedback decoupling controllers which are robust to *small* perturbations by means of an eigenvector approach. Noteworthy in this work is the use of geometrical interpretations [Ref. 16] to analysis of the allowable closed loop design space as functions of open loop parameters.

Andry, Shapiro and Chung [Ref. 17] extended the analytical work in eigenstructure assignment by investigating the constrained output feedback case. By assigning fixed zeroes to specified feedback gain elements, they noted that engineering flexibility is gained by the designer in terms of being able to compute a *spectrum* of satisfactory controllers. This enables one to compare the entire range of controllers (*full state feedback through constrained output feedback*) with respect to performance, cost, and reliability.

Magni and Herail [Ref. 18] presented methods which invoke disturbance decoupling via output feedback. These researchers used the design freedom of eigenvector specification to minimize the norm of the transfer matrix between the particular disturbance and the controlled state variables. This in effect attenuated the particular disturbance in the resulting system response. They applied these results to an aircraft gust alleviation (1 - cosine) problem with some success.

Eastman and Bossi [Ref. 19] generalized the linear quadratic gaussian eigenvalue placement technique of Solheim [Ref. 20] to include specification of the allowable closed loop eigenvectors. An iterative algorithm, eigenvector specification was accomplished by manipulating the geometric structure of the Ricatti matrix at each iterative stage.

Numerical methods for computing robust state feedback gains as explicit functions of the closed loop eigenstructure were presented by Kautsky, Nichols, and Van Dooren [Ref. 21]. Noteworthy in this work is the exploitation of an established matrix factorization, the singular value decomposition (*SVD*) [Ref. 22]. Through the use of *SVD* these authors arrived at an expression for full state feedback gains as an explicit function of the closed loop eigenstructure. In this thesis, an extension of their analysis to the output feedback case involving the closed loop left (or dual) eigenvectors will be presented.

Of the several analytical works which focus on eigenstructure assignment for linear dynamic systems, mention should be made of the paper by Sobel and Shapiro [Refs. 23,24]. A two part work, Part I [Ref. 23] presents eigenstructure assignment theory in a tutorial fashion. Part II [Ref. 24] presents informative applications to flight control design via output and constrained output feedback controllers which were to meet specified modal structures.

Several recent authors have continued the research of applying eigenstructure assignment theory to linear state feedback design. Mielke, Carraway, and Marefat

[Ref. 25] presented an interactive design algorithm using the classic results of Srinathkumar [Ref. 13] and [Ref. 10] while Liebst and Garrad [Ref. 26] applied the work of Andry, et. al. [Ref. 17] to aircraft flutter control and gust alleviation problems. White and Speyer [Ref. 27] presented innovative results of the application of eigenstructure assignment to failure detection filters, a type of observer.

Recent analytical research in eigenstructure theory has centered on subspace characterizations of the allowable sets of closed loop eigenvectors. Fletcher, Kautsky, Kolka, and Nichols [Ref. 28] arrived at explicit expressions of feedback gains as functions of right and left eigenstructures. This paper provided the initial motivation to examine the increased eigenvector constraints imposed by output feedback in terms of the dual constraints on the allowable left eigenvectors. Sogaard-Anderson, Trostmann, and Conrad [Ref. 29] characterized the sets of allowable right and left eigenvector sets in terms of residual subspaces defined by the matrix residuals associated with the desired closed loop eigenvalues. This work however, did not explicitly examine the left eigenvector sets as members of a particular subspace. This thesis will examine such a membership.

In this thesis, research objectives were accomplished in two phases. The initial phase concentrated on the coding of an interactive eigenstructure design algorithm (*EIGENS*) for linear dynamic systems with no restrictions as to state or output feedback [Ref. 30]. Applications of the algorithm to design of robust decoupling controllers were successfully performed on the CH-47 helicopter. Further application to design of a decoupling output controller for the L-1011 transport was also accomplished through execution of the *EIGENS* algorithm. The latter phase of the research entailed developing ***geometrical*** interpretations of the restrictions of choosing allowable right and left eigenvectors, innovating a new application of eigenstructure assignment to ***reconfiguring digital flight controllers***, and an application of this new concept to the F/A-18 tactical aircraft. Let us now turn to developing the necessary eigenstructure theory for linear dynamic systems for use throughout the thesis.

III. EIGENSTRUCTURE THEORY FOR LINEAR TIME INVARIANT SYSTEMS

A. PRELIMINARIES

1. The Singular Value Decomposition

Matrix and vector notation used throughout this thesis is noted in APPENDIX A. The matrix singular value decomposition and the matrix pseudo - inverse are reviewed below.

Consider a matrix B, where $B \in \mathscr{R}^{nxm}$. The singular value decomposition (SVD) of B is defined as [Ref. 22],

$$B = U_b \Sigma_b V_b{}^T \qquad \text{(eqn 3.1)}$$

where U_b and $V_b{}^T$ are orthogonal matrices of order *n and m* respectively. The matrix Σ_b is a diagonal matrix of the singular values of the matrix B.

$$\Sigma_b = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_p) \qquad \text{(eqn 3.2)}$$

where *p* is min(n,m).

Additionally, the singular values, σ_i, are commonly defined in terms of the spectrum of BB^T such that,

$$\sigma_i(B) = \sqrt{\lambda_i(BB^T)} \qquad \text{(eqn 3.3)}$$

In the arguments which arise in this thesis, the following block form of equation 3.1 will be used extensively for full rank matrices where $n \geq m$.

$$B = \{U_{bo} \; U_{b1}\} \begin{bmatrix} Z_b \\ \Phi \end{bmatrix} \qquad \text{(eqn 3.4)}$$

where Φ denotes a null matrix and Z_b is defined by

$$Z_b = \Sigma_b V_b{}^T \qquad \text{(eqn 3.5)}$$

Since it is assumed that B is of full rank, note that Σ_b is now a square mxm matrix where the null blocks of equation 3.2 have been disregarded. In all the following discussions, Σ_b will be assumed to be square. The linear space dimensions of the block matrices are shown below.

$$U_{bo} \in \mathscr{R}^{nxm} \qquad \text{(eqn 3.6)}$$

$$U_{b1} \in \mathscr{R}^{nx(n-m)} \qquad \text{(eqn 3.7)}$$

$$Z_b \in \mathscr{R}^{mxm} \qquad \text{(eqn 3.8)}$$

$$\Phi \in \mathscr{R}^{(n-m)xm} \qquad \text{(eqn 3.9)}$$

Note that for non - singular square matrices, U_{b1} does not exist and equation 3.4 reduces to

$$B = U_b Z_b = U_{bo} Z_b \qquad \text{(eqn 3.10)}$$

Useful identities associated with the matrix singular value decomposition are noted below.

$$\sigma_1 \geq \ldots \sigma_r > \sigma_{r+1} = \ldots = 0 \qquad \text{(eqn 3.11)}$$

where r = rank(B).

$$\sigma_1(B) \equiv \sigma_{max}(B) \qquad \text{(eqn 3.12)}$$

$$\sigma_r(B) \equiv \sigma_{min}(B) \qquad \text{(eqn 3.13)}$$

$$\|B\|_2 \equiv \sigma_1(B) \qquad \text{(eqn 3.14)}$$

For square matrices, where $B \in \mathscr{R}^{nxn}$, the two norm condition number of the matrix B, $\kappa_2(B)$, is defined as,

$$\kappa_2(B) = \sigma_1(B) / \sigma_n(B) \qquad \text{(eqn 3.15)}$$

2. The Matrix Pseudo Inverse

The matrix pseudo inverse, often referred to as the generalized inverse, is defined in the least squares sense in the following way. For the matrix $B \in \mathscr{R}^{nxm}$, the pseudo inverse, B^+, is the unique Frobenius-norm solution to,

$$\min \| BB^+ - I_n \|_F \qquad \text{(eqn 3.16)}$$

Additionally, B^+ satisfies the classic Moore - Penrose conditions,

$$BB^+B = B \qquad \text{(eqn 3.17)}$$

$$B^+BB^+ = B^+ \qquad \text{(eqn 3.18)}$$

$$(BB^+)^T = B^+B \qquad \text{(eqn 3.19)}$$

$$(B^+B)^T = B^+B \qquad \text{(eqn 3.20)}$$

In this thesis, the singular value decomposition will be used to define the matrix pseudo inverse in the following way [Ref. 31],

$$B^+ = V_b \Sigma_b^{-1} U_{bo}^T \qquad \text{(eqn 3.21)}$$

where,

$$B^+ \in \mathscr{R}^{mxn} \qquad \text{(eqn 3.22)}$$

It can be shown that equation 3.22 satisfies the Moore - Penrose conditions. Proof of equality for equation 3.17 is shown below. Substituting the expression from equation 3.21 into equation 3.17 yields,

$$BB^{+}B = (U_{bo}\Sigma_b V_b{}^T)(V_b \Sigma_b{}^{-1} U_{bo}{}^T)(U_{bo}\Sigma_b V_b{}^T) \qquad \text{(eqn 3.23)}$$

Since,

$$V_b{}^T V_b = I_m \qquad \text{(eqn 3.24)}$$

and,

$$U_{bo}{}^T U_{bo} = I_m \qquad \text{(eqn 3.25)}$$

then equation 3.23 reduces to,

$$BB^{+}B = U_{bo}\Sigma_b(I_m)\Sigma_b{}^{-1} I_m \Sigma_b V_b{}^T \qquad \text{(eqn 3.26)}$$

or,

$$BB^{+}B = U_{bo}\Sigma_b \Sigma_b{}^{-1}\, \Sigma_b V_b{}^T \qquad \text{(eqn 3.27)}$$

Further, since

$$\Sigma_b \Sigma_b{}^{-1} = I_m \qquad \text{(eqn 3.28)}$$

then one has the final result,

$$BB^{+}B = U_{bo}\Sigma_b V_b{}^T = B \qquad \text{(eqn 3.29)}$$

B. THE GENERALIZED FEEDBACK EXPRESSION FOR LINEAR TIME INVARIANT SYSTEMS

Kautsky, Nichols, and Van Dooren [Ref. 21] published the development of an expression for ***state feedback*** gains as an explicit function of a desired closed loop eigenstructure. The following general derivation follows closely that of Kautsky et.al. by expanding their results to include the output feedback case. *Generalized* in this thesis is interpreted to include output feedback with the existence of feed forward loops in the system. Full state feedback or output feedback without feed forward then become *specific* cases of the final expression.

Consider the linear *MIMO* system comprised of *n states, m inputs, and l outputs* with output feedback,

$$\dot{x} = Ax + Bu \qquad \text{(eqn 3.30)}$$

$$y = Cx + Du \qquad \text{(eqn 3.31)}$$

$$u = Fy \qquad \text{(eqn 3.32)}$$

The classical eigenvalue placement problem written in vector form arises from combining equations 3.30, 3.31, and 3.32 in the following fashion. Combining equations 3.31 and 3.32 yields,

$$u = FCx + FDu \qquad \text{(eqn 3.33)}$$

and rearranging yields,

$$(I_m - FD)u = FCx \qquad \text{(eqn 3.34)}$$

Assuming $(I_m - FD)$ is not singular one pre multiplies equation 3.34 by $(I_m - FD)^{-1}$ yielding,

$$u = (I_m - FD)^{-1}FCx \qquad \text{(eqn 3.35)}$$

Substituting equation 3.35 into equation 3.30, one has

$$\dot{x} = Ax + B\{(I_m - FD)^{-1}FC\}x \qquad \text{(eqn 3.36)}$$

Upon application of an appropriate similarity transformation, equation 3.36 may be written in terms of the closed loop system eigenvectors, x_i, and eigenvalues, λ_i [Ref. 2], as follows,

$$\lambda_i x_i = Ax_i + B\{(I_m - FD)^{-1}FC\}\, x_i \qquad \text{(eqn 3.37)}$$

In matrix form, equation 3.37 becomes,

$$X\Delta X^{-1} = A + B(I_m - FD)^{-1}FC \qquad \text{(eqn 3.38)}$$

where Δ is the diagonal matrix of closed loop eigenvalues and it is implicitly assumed that the eigenvector matrix, X, is composed of linearly independent columns.

Subtracting A from both sides of equation 3.38 yields,

$$B(I_m - FD)^{-1}FC = X\Delta X^{-1} - A \qquad \text{(eqn 3.39)}$$

Now by exploiting the singular value decompositions of *B and C* where,

$$B = U_{bo}\Sigma_b V_b^T = U_{bo}Z_b \qquad \text{(eqn 3.40)}$$

and,

$$C = U_c\Sigma_c V_c^T = U_{co}Z_c \qquad \text{(eqn 3.41)}$$

equation 3.39 becomes,

$$U_{bo}Z_b(I_m - FD)^{-1}FU_{co}Z_c = X\Delta X^{-1} - A \qquad \text{(eqn 3.42)}$$

Appropriate pre and post matrix multiplication of equation 3.42 yields,

$$(I_m - FD)^{-1}F = Z_b^{-1}U_{bo}^T(X\Delta X^{-1} - A)Z_c^{-1}U_{co}^T \qquad \text{(eqn 3.43)}$$

and an additional pre matrix multiplication of equation 3.43 by by (I_m - FD) results in,

$$F = (I_m - FD)Z_b^{-1}U_{bo}^T(X\Delta\, X^{-1} - A)Z_c^{-1}U_{co}^T \qquad \text{(eqn 3.44)}$$

Let us now define the eigenstructure matrix *L*, as,

$$L = (X\Delta X^{-1} - A) \qquad \text{(eqn 3.45)}$$

Rearranging equation 3.44 with the substitution of equation 3.45 yields,

$$F + FD(Z_b^{-1}U_{bo}LZ_c^{-1}U_{co}^T) = Z_b^{-1}U_{bo}^TLZ_c^{-1}U_{co}^T \qquad \text{(eqn 3.46)}$$

Now define *Q as*,

$$Q = Z_b^{-1}U_{bo}^TLZ_c^{-1}U_{co}^T \qquad \text{(eqn 3.47)}$$

Then equation 3.46 becomes,

$$F(I_m + DQ) = Q \qquad \text{(eqn 3.48)}$$

and appropriate post matrix multiplication of 3.48 yields the following general expression for the least square output feedback gain matrix, *F*, as a function of a desired closed loop eigenstructure, $X\Delta X^{-1}$.

$$F = Q(I_m + DQ)^{-1} \qquad \text{(eqn 3.49)}$$

Note that the right hand side of equation 3.49 is a function of the pseudo inverses of B and C^T expressed in equation 3.47 . The feedback gain matrix, F, therefore is computed based on these least square solutions. As such, F will invoke the desired closed loop eigenstructure *X* and Δ, in the least square sense. For systems where the number of states, inputs, and outputs are identical, then the solution becomes exact. As the number of inputs and outputs vary, then the solution becomes inexact by the nature of the pseudo inverse. This is a numerical interpretation of the classic restrictions on the desired eigenstructure presented by Srinathkumar [Ref. 11].

A proof of this result using the matrix equations presented in this thesis is shown in Appendix B. Equation 3.49 is therefore more correctly termed the least square solution for the feedback gains, F. It is this solution which was used for the results presented in this thesis.

Up to this point, no restrictions have been placed on the desired right eigenvector matrix, X, other than the right eigenvectors must form a linearly independent set. Additional restrictions on the right eigenvectors are shown in the following development. Rewriting equation 3.39 in block form yields,

$$\begin{bmatrix} U_{bo} & U_{b1} \end{bmatrix} \begin{bmatrix} Z_b \\ \Phi \end{bmatrix} (I_m - FD)^{-1} FC = X \Lambda X^{-1} - A \qquad \text{(eqn 3.50)}$$

Pre multiplying both sides of 3.50 by, $\{U_{bo}\ U_{b1}\}^T$ one has,

$$\begin{bmatrix} Z_b \\ \\ \Phi \end{bmatrix} (I_m - FD)^{-1} FC = \begin{bmatrix} U_{bo}^T \\ \\ U_{b1}^T \end{bmatrix} (X \Lambda X^{-1} - A) \qquad \text{(eqn 3.51)}$$

Upon multiplying equation 3.51 by -1, one has the following relationship resulting from the lower block,

$$\Phi = U_{b1}^T (A - X \Lambda X^{-1}) \qquad \text{(eqn 3.52)}$$

Equation 3.52 is the matrix relationship [Ref. 21] from which the subspace constraint on each right eigenvector may be obtained. This subspace constraint is repeated below,

$$x_i \in N\{U_{b1}^T (A - \lambda_i I_n)\} \qquad \text{(eqn 3.53)}$$

where N denotes null space. Equation 3.53 is the only subspace restriction on the desired linearly independent right eigenvectors for the full state feedback case.

The dimension, *d*( ), of this particular subspace hereafter referred to as the ***right eigenspace***, can be defined in the following way. Denoting Ker P as the kernel or null space and Im P as the image or range space of a mapping operator P; one has the identity [Ref. 16]

$$d(X) = d(\text{Ker } P) + d(\text{Im } P) \qquad \text{(eqn 3.54)}$$

where Im P = PX. In this case, X is the *n - dimensional* state space spanned by the linearly independent right eigenvectors. Solving equation 3.54 for d(Ker P), one has,

$$d(\mathrm{Ker}\ P) = n - d(\mathrm{Im}\ P) \qquad \text{(eqn 3.55)}$$

Since d(Im P) is also defined as the rank of P, one may rewrite equation 3.55 as,

$$d(\mathrm{Ker}\ P) = n - \mathrm{rank}(P) \qquad \text{(eqn 3.56)}$$

Substituting the right eigenspace operator for P now results in a definition for the dimension of the right eigenspace.

$$d(\mathrm{Ker}\{U_{b1}{}^T(A - \lambda_i I_n)\}) = n - \mathrm{rank}\{U_{b1}{}^T(A - \lambda_i I_n)\} \qquad \text{(eqn 3.56)}$$

and since,

$$\mathrm{rank}\{U_{b1}{}^T(A - \lambda_i I_n)\} = n - m \qquad \text{(eqn 3.58)}$$

for full rank control (rank(B) = m) matrices, one has the following result,

$$d(N\{U_{b1}{}^T(A - \lambda_i I_n)\}) = m \qquad \text{(eqn 3.59)}$$

where Ker has been replaced with the original notation for the null space operator, N.

Therefore, the allowable right eigenvector x_i, must be a member of an *m* dimensional null subspace defined by equation 3.53 . Note further that for a rank degenerate control matrix, B, the subspace likewise becomes dimensionally degenerate.

It will now be shown that in the presence of output feedback, an additional restriction of the left eigenvectors results from a similar analysis. In order to show this additional restriction, let us initially assume the feed forward matrix, ***D*** to be identically null which is the case of output feedback without the addition of transmission or blocking zeroes. There is no loss of generality with this assumption. In this case, equation 3.48 reduces to,

$$F = Q = Z_b{}^{-1} U_{b1}{}^T L Z_c{}^{-1}\ U_{co}{}^T \qquad \text{(eqn 3.60)}$$

and substituting the expression for L yields,

$$F = Z_b^{-1} U_{bo}^T (X \Delta X^{-1} - A)\, Z_c^{-1} U_{co}^T \qquad \text{(eqn 3.61)}$$

with the constraint that each right eigenvector is a member of a specific null space as expressed by equation 3.53 .

Pausing briefly, note that if one assumed all the states were available for feedback, where the output matrix, C, becomes I_n, equation 3.61 reduces to the expression shown previously [Ref. 21].

$$F = Z_b^{-1} U_{bo}^T (X \Delta X^{-1} - A) \qquad \text{(eqn 3.62)}$$

Comparing equations 3.62, 3.61, and 3.48, one notes the increased cost in computing F when proceeding from full state feedback to output feedback with feed forward control.

Let us now return to analyzing the impact of output feedback on the left eigenvectors by transposing equation 3.38 with D identically null.

$$-C^T F^T B^T = A^T - X^{-T} \Delta^T X^T \qquad \text{(eqn 3.63)}$$

Factorization of C^T via singular value decomposition yields,

$$-[U_{cto}\ U_{ct1}] \begin{bmatrix} \Sigma_{ct} V_{ct} \\ \Phi \end{bmatrix} F^T B^T = A^T - X^{-T} \Delta^T X^T \qquad \text{(eqn 3.64)}$$

Pre matrix multiplication of equation 3.63 by $\{U_{cto}\ U_{ct1}\}^T$ yields,

$$-\begin{bmatrix} \Sigma_{ct} V_{ct} \\ \Phi \end{bmatrix} F^T B^T = \begin{bmatrix} U_{cto}^T \\ U_{ct1}^T \end{bmatrix} (A^T - X^{-T} \Delta^T X^T) \qquad \text{(eqn 3.65)}$$

and from the lower block of equation 3.65,

$$\Phi (F^T B^T) = \Phi = U_{ct1}^T (A^T - X^{-T} \Delta^T X^T) \qquad \text{(eqn 3.66)}$$

Now using the following definition [Ref. 22] of the left eigenvector matrix, T,

$$T = X^{-T} \qquad \text{(eqn 3.67)}$$

equation 3.66 becomes,

$$\Phi = U_{ctl}{}^{T}(A^{T} - T\Delta^{T}T^{-1}) \qquad \text{(eqn 3.68)}$$

and since,

$$\Delta = \Delta^{T} \qquad \text{(eqn 3.69)}$$

the final matrix expression for equation equation 3.66 becomes,

$$\Phi = U_{ctl}{}^{T}(A^{T} - T\Delta T^{-1}) \qquad \text{(eqn 3.70)}$$

Post multiplying equation 3.70 by T yields,

$$\Phi T = U_{ctl}{}^{T}(A^{T}T - T\Delta) \qquad \text{(eqn 3.71)}$$

and in vector form,

$$\{0\} = U_{ctl}{}^{T}(A^{T} - \lambda_i I_n)t_i \qquad \text{(eqn 3.72)}$$

and hence for the output feedback case, the *ith* left eigenvector must be a member of the following null space,

$$t_i \in N\{U_{ctl}{}^{T}(A^{T} - \lambda_i I_n)\} \qquad \text{(eqn 3.73)}$$

It can be similarly shown that the dimension of this space, defined as the *left eigenspace*, is,

$$d\{N(U_{ctl}{}^{T}\{A^{T} - \lambda_i\})\} = l \qquad \text{(eqn 3.74)}$$

when the output matrix C is full rank.

To summarize for the output feedback case without feed forward control, the feedback gain matrix, *F*, is explicitly defined by equation 3.61 with equations 3.53 and 3.73 placing constraints on the right and left eigenvectors respectively. Additionally note that for the full state feedback case where $C = I_n$, there is no restriction on the left eigenvector, t_i since U_{ct1} does not exist and any left vector would satisfy equation 3.71 . Similarly note that for the full state feedback case where rank(B) is *n*, *or for the system with n inputs*, there is also no subspace constraint on the right eigenvector. Therefore for the full state feedback case where $m = n$, one can choose any *n linearly independent* set of closed loop right eigenvectors for the desired closed loop modal structure.

One now has at their disposal, constrained explicit expressions for calculating static feedback gains which invoke a desired closed loop eigenstructure in the least square sense. Table 1 summarizes the expressions for the feedback gains along with appropriate eigenvector constraints for each feedback scheme.

TABLE 1

FEEDBACK EXPRESSIONS AND EIGENVECTOR CONSTRAINTS

Full State Feedback

$$F_{sf} = Z_b^{-1} U_{bo}^T (X \Lambda X^{-1} - A)$$
$$x_i \in N\{U_{b1}^T (A - \lambda_i I_n)\}$$

Output Feedback without Feedforward Control Loop

$$F_{of} = Z_b^{-1} U_{bo}^T (X \Lambda X^{-1} - A) Z_c^{-1} U_{co}^T$$
$$x_i \in N\{U_{b1}^T (A - \lambda_i I_n)\}$$
$$t_i \in N\{U_{ct1}^T (A^T - \lambda_i I_n)\}$$

Output Feedback with Feedforward Control Loop

$$F_{off} = F_{of}(I_m + D F_{of})^{-1}$$
$$x_i \in N\{U_{b1}^T (A - \lambda_i I_n)\}$$
$$t_i \in N\{U_{ct1}^T (A^T - \lambda_i I_n)\}$$

As a preface to the ensuing discussion, let us examine the numerical and linear space definitions of the term ***null space.*** Null space membership, as previously defined by equation 3.52 for example, will not be exactly satisfied due to finite precision arithmetic and system modelling uncertainty. Defining a null vector as ε and a null matrix as $\mathbf{E}$, where

$$\|\varepsilon\|_2 << 1 \qquad \text{(eqn 3.75)}$$

$$\sigma_1(\mathrm{E}) << 1 \qquad \text{(eqn 3.76)}$$

allows one to more clearly define numerical null space membership. Therefore, let us account for computational precision by defining the right and left null vectors, ε_{ri} and ε_{li} as follows.

$$\varepsilon_{ri} = U_{b1}{}^T(A - \lambda_i I_n)\, x_i \qquad \text{(eqn 3.77)}$$

$$\varepsilon_{li} = U_{ct1}{}^T(A^T - \lambda_i I_n)\, t_i \qquad \text{(eqn 3.78)}$$

Further, assume these vectors span respective right and left null spaces,

$$\mathrm{E}_r = \{\varepsilon_{r1}\ \varepsilon_{r2} \cdots \varepsilon_{rn}\} \qquad \text{(eqn 3.79)}$$

$$\mathrm{E}_l = \{\varepsilon_{l1}\ \varepsilon_{l2} \cdots \varepsilon_{ln}\} \qquad \text{(eqn 3.80)}$$

where,

$$\mathrm{E}_r \subset \mathscr{R}^{n-\mathrm{rank}(B)} \qquad \text{(eqn 3.81)}$$

$$\mathrm{E}_l \subset \mathscr{R}^{n-\mathrm{rank}(C^T)} \qquad \text{(eqn 3.82)}$$

Finally note that for full rank B and C^T matrices, the dimensions of the right and left null spaces are therefore,

$$d(E_r) = n - m = p \qquad \text{(eqn 3.83)}$$

$$d(E_l) = n - l = q \qquad \text{(eqn 3.84)}$$

These dimensional definitions are used in the following discussion to more clearly define the algebraic constraints on the allowable eigenstructure. Let us now continue with such a discussion.

C. RIGHT AND LEFT EIGENSPACE: A GEOMETRIC INTERPRETATION

The algebraic subspace constraints noted previously can be difficult to envision for large order systems. In addition, the added restrictions on the allowable right and/or left eigenvectors as one progresses from the full state feedback - full rank control matrix system to a constrained output feedback - rank degenerate control or output case has intuitive geometrical properties. It is the geometrical implications imposed by the algebraic constraints which motivate, in part, the following geometrical analysis.

The lattice diagram [Ref. 16] has proved to be a useful device to depict geometrical relationships between linear vector spaces. The following geometrical relationships are implied by Figure 3.1.

$$E_r^p \subset \mathscr{R}^m \subset X^n \qquad \text{(eqn 3.85)}$$

$$d(E_r^p) < d(\mathscr{R}^m) < d(X^n) \qquad \text{(eqn 3.86)}$$

where the symbol $\subset$ denotes subspace. Equation 3.86 states formally that the dimension of $\mathscr{R}^m$ is less than the dimension of X^n and the dimension of E_r^p is less than that of $\mathscr{R}^m$.

Let us now assume, that X^n represents the state space of a controllable linear time invariant system such that,

$$X^n = \mathscr{R}_1{}^m + \mathscr{R}_2{}^m + \dots \mathscr{R}_n{}^m \qquad \text{(eqn 3.87)}$$

As there are non-unique state variable representations for a linear time invariant system, there may be several linearly independent eigenvector sets which are able to span the state space. However if a vector is chosen from each of the m-dimensional spaces $\mathscr{R}_i{}^m$ to form spanning sets, one has selected particular eigenvectors which are associated with particular eigenvalues. It is this selection process which will be invoked by the feedback gain equations of Table I.

Assume further that for each subspace, $\mathscr{R}_i{}^m$, there exists a mapping operator, N_{ri}, which maps a vector from $\mathscr{R}_i{}^m$ into $\varepsilon_{ri}{}^p$, a numerical *null vector*.

$$\varepsilon_{ri}{}^p = N_{ri}x_i \qquad \text{(eqn 3.88)}$$

Further assume that any p of these null vectors span a right null space, $E_r{}^p$, where,

$$E_r{}^p = \{\varepsilon_{r1}{}^p \ \ \varepsilon_{r2}{}^p \ \dots \ \varepsilon_{rn}{}^p\} \qquad \text{(eqn 3.89)}$$

where equation 3.89 has included all n right null vectors.

The subsequent lattice diagram, denoted as the Right State Space Lattice is shown in Figure 3.2.

Clearly, N_{ri} can be taken to be the right null space operator defined previously as,

$$N_{ri} = U_{b1}{}^T(A - \lambda_i I_n) \qquad \text{(eqn 3.90)}$$

and Figure 3.2 geometrically depicts the right eigenspace constraints posed by state feedback. The dual or left state space lattice is similarly constructed and is shown in Figure 3.3.

The final tool in constructing the total lattice diagram is to geometrically link the relationship between Figures 3.2 and 3.3. Let us define an operator P, such that,

$$P = X^{-T}X^{-1} \qquad \text{(eqn 3.91)}$$

Therefore, P maps the right eigenvector matrix, X, into the left eigenvector matrix, T, as follows,

$$T = PX = X^{-T}X^{-1}X \qquad \text{(eqn 3.92)}$$

and one may now map the allowable n right eigenvectors into their dual left eigenvector space.

Likewise, we desire to develop an operator which will map the right null vectors represented by equation 3.89 into their dual left null space vectors represented by $\mathbf{E}_l$.

For each ith null vector, one would then have,

$$J_i \varepsilon_{ri} = \varepsilon_{li} \qquad \text{(eqn 3.93)}$$

In terms of the ith right and left eigenvector, equation 3.93 becomes,

$$J_i \mathbf{N}_{ri} x_i = \mathbf{N}_{li} t_i \qquad \text{(eqn 3.94)}$$

Let us now define a vector s_i as the ith column of the I_n identity matrix. One may therefore write an expression for t_i as follows,

$$t_i = Ts_i \qquad \text{(eqn 3.95)}$$

Substituting the expression for T from equation 3.92 yields,

$$t_i = PXs_i = Px_i \qquad \text{(eqn 3.96)}$$

and therefore equation 3.94 becomes,

$$J_i \mathbf{N}_{ri} x_i = \mathbf{N}_{li} P x_i \qquad \text{(eqn 3.97)}$$

Solving for J_i yields,

$$J_i = (\mathbf{N}_{li} P x_i)(\mathbf{N}_{ri} x_i)^{+} \qquad \text{(eqn 3.98)}$$

In order to verify that equation 3.98 is correct, one must show that J_i indeed maps ε_{ri} into ε_l.

Performing the mapping yields,

$$(N_{li}Px_i)(N_{ri}x_i)^+(N_{ri}x_i) \qquad \text{(eqn 3.99)}$$

Noting that $(N_{ri}x_i)^+(N_{ri}x_i)$ is equal to scalar unity (1.0), then the mapping of equation 3.99 yields,

$$N_{li}Px_i(1) = \varepsilon_{li} \qquad \text{(eqn 3.100)}$$

and J_i maps ε_{ri} into ε_{li}. In this way, one may map the 'n' ε_{ri} null vectors into the respective 'n' left null vectors, ε_{li}. One may now depict the total space lattice diagram shown in Figure 3.4..

The geometrical constraints on the allowable right and left eigenvectors shown in Figure 3.4 can be characterized as follows. The dimensions of the right and left null spaces, $\mathbf{E}_r$ and $\mathbf{E}_l$ are directly related to the rank of the null space operators N_r and N_l. For example, as the rank of N_r decreases, the dimension of $\mathbf{E}_r$ decreases. Since the right eigenvector x_i must be mapped into ε_r by N_{ri}, the mapping becomes more geometrically restrictive. In this way, an increasing geometric constraint is placed on the right eigenvector x_i. The dual is true for the left eigenvector t_i. One may state the geometrical constraints in the following way.

> Lemma 3.1: The expressions of Table I explicitly define static feedback gains which will invoke a closed loop right eigenstructure, $X\Lambda X^{-1}$, iff the right eigenvectors, x_i, which *reside* in $\mathscr{R}_i^m$ map into the right null space $\mathbf{E}_r$ via the respective null space operator N_{ri}. The dual vectors, t_i must likewise *reside* in L_{il} and map into $\mathbf{E}_l$ via N_{li}.

The concept of right and left null spaces comes about when one numerically converges on an allowable eigenvector by allowing the vector to rotate until the respective null vector, ε, becomes small. In this way one numerically converges on eigenvectors which are members of $\mathscr{R}_i^m$ or $\mathscr{R}_i^l$ by minimizing a norm of ε_r or ε_l. This solution technique was used in this thesis and provided the prime motivation for these geometrical discussions. Figure 3.4 has set the stage for geometrical analysis of all the feedback cases to be presented. Let us begin with the state feedback case as it is the least restrictive with respect to allowable eigenstructure.

Case I: Full State Feedback ($m = n$)

In this case the diagram reduces to a single node for each eigenspace. There are no restrictions on the desired vectors other than linear independence and the feedback gains which will invoke such a closed loop structure are computed via equation 3.62 .

Case II: Full State Feedback ($m < n$)

For this case, one has an initial restriction on the right eigenvectors but none on the left. Therefore the total lattice diagram consists of Figure 3.5 with subspace inclusions for the right eigenspace. Note the dimension of $\mathbf{E}_r^P$ defined previously by equation 3.83 . By comparing Figures 3.5 and 3.6, one notes the first level of restriction with regard to the allowable closed loop eigenstructure.

Case III: Output Feedback ($m = n, l = n$)

Case III is geometrically identical to Case I. The only restrictions are that the desired closed loop right and left eigenvectors form a bi-orthogonal set. Figure 3.5 depicts the appropriate geometry.

Case IV: Output Feedback ($m = n, l < n$)

For this case, the restrictions lie only with the left eigenvectors. The lattice diagram is the dual of Figure 3.6 and is is shown in Figure 3.7.

This concludes the decoupled constraint cases and exemplifies the ease of solution for the gain matrix which will invoke desired closed loop eigenstructures. The coupled cases arise for systems employing output feedback where both the number of inputs and observations are less than the order of the state space.

Case V: Output Feedback ($m < n, l < n, m = l$)

The lattice diagram for this case is constructed by linking Figures 3.6 and 3.7 with the mapping operator P and is depicted in Figure 3.8. Note from equations 3.83 and 3.84 that,

$$d(\mathbf{E}_r) = d(\mathbf{E}_l) \qquad \text{(eqn 3.101)}$$

and therefore, the null spaces are equidimensional.

Case VI: Output Feedback ($m < n, l < n, m < l$)

The lattice diagram of Figure 3.8 remains for the remaining cases. The subtle distinction between Cases V and Case VI however, is that the dimensions of the left and right null space are not identical. For this case,

$$d(E_r) > d(E_l) \qquad \text{(eqn 3.102)}$$

which implies geometrically that the right eigenspace constraint is more easily satisfied due to the ***higher dimension*** of the right null space.

Case VII: Output Feedback ($m < n, l < n, l < m$)

Case VII is the dual of Case VI, since

$$d(E_l) > d(E_r) \qquad \text{(eqn 3.103)}$$

Equation 3.103 implies that the left eigenspace constraints are more easily satisfied than those of the right eigenspace due to the ***higher dimension*** of the left null space. This of course is the *dual* of Case VI in that the major computational expense will involve converging on allowable right eigenvector sets. Cases VI and VII are clearly computationally more difficult to solve than the previous cases. The quantitative interplay between the right and left eigenvector sets as functions of the open loop parameters remains to be clearly formulated. The attempt here was to show the qualitative interdependency by analyzing what was found to be true during the course of the research. It will be shown later in this thesis using two specific examples of Case VI, that the qualitative interpretations noted above are correct.

This concludes the theoretical treatment of eigenstructure assignment as presented in this thesis. Prior to discussing some applications of the theory, let us reexamine equations 3.77 and 3.78 As noted previously, null space membership of the desired closed loop eigenvectors has been the central theme in much of the recent discussion. One must, therefore, numerically devise a scheme which guarantees such membership. In that the term *null space* implies convergence to a zero measure, a candidate for such a scheme would clearly involve a minimization. Minimizing a norm of the right null vector ε_{ri} or left null vector ε_{li} for example, would guarantee such convergence. The mechanics of such a minimization is the topic of the final chapter prior to presenting results of applying eigenstructure assignment to some existing control problems.

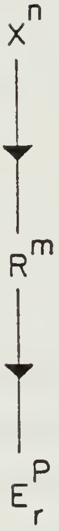


Figure 3.1 Basic Lattice Diagram.

$$\begin{tikzcd}
 & & X^n \arrow[dll] \arrow[d] \arrow[drr] & & \\
R_1^m \arrow[drr, "Nr_1"'] & & R_2^m \arrow[d, "Nr_2"'] & \cdots & R_n^m \arrow[dll, "N_{r_n}"] \\
 & & E_r^p & \cdots &
\end{tikzcd}$$

Figure 3.2 Right State Space Lattice Diagram.

$$\begin{tikzcd}
 & T^n \arrow[dl] \arrow[d] \arrow[drr] & & \\
L_1^{\ell} \arrow[dr, "N_{\ell_1}"'] & L_2^{\ell} \arrow[d, "N_{\ell_2}"] & \cdots & L_n^{\ell} \arrow[dll, "N_{\ell_n}"] \\
 & E_{\ell}^{Q} & &
\end{tikzcd}$$

Figure 3.3 Left State Space Lattice Diagram.

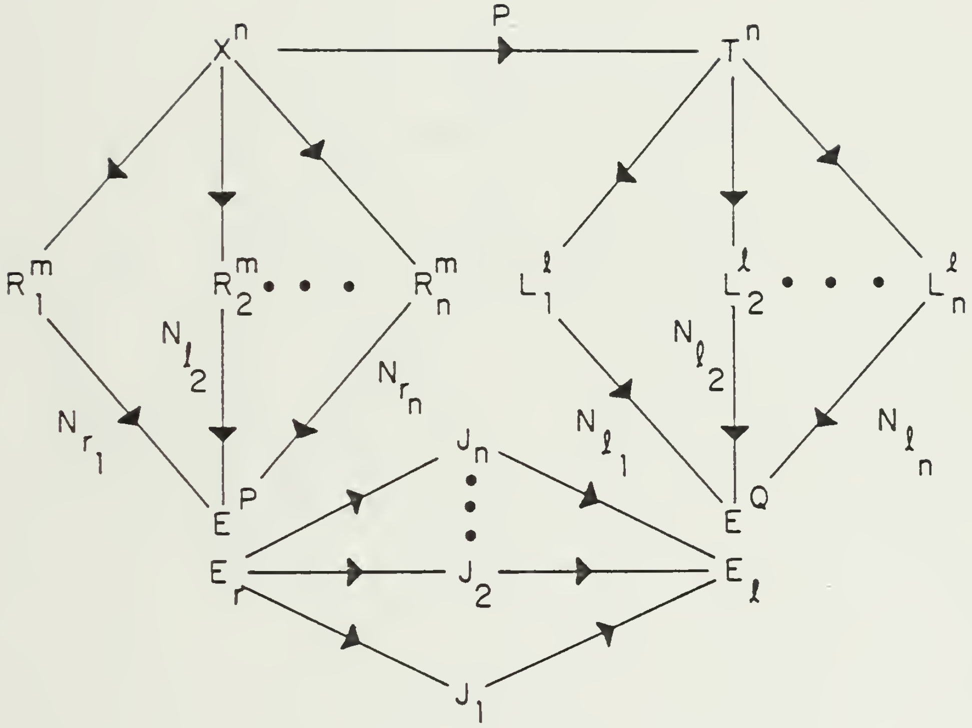


Figure 3.4 Total Space Lattice Diagram.

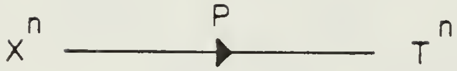


Figure 3.5 Case I: Full State Feedback (m = n) Lattice Diagram.

$$\begin{tikzcd}
 & X^n \arrow[dl] \arrow[d] \arrow[drr] \arrow[rrr, "P"] & & & T^n \\
R_1^m \arrow[dr, "N_{r_1}"'] & R_2^m \arrow[d, "N_{r_2}"'] & \cdots & R_n^m \arrow[dll, "N_{r_n}"] & \\
 & E_r^P & & &
\end{tikzcd}$$

Figure 3.6 Case II: Full State Feedback (m < n) Lattice Diagram.

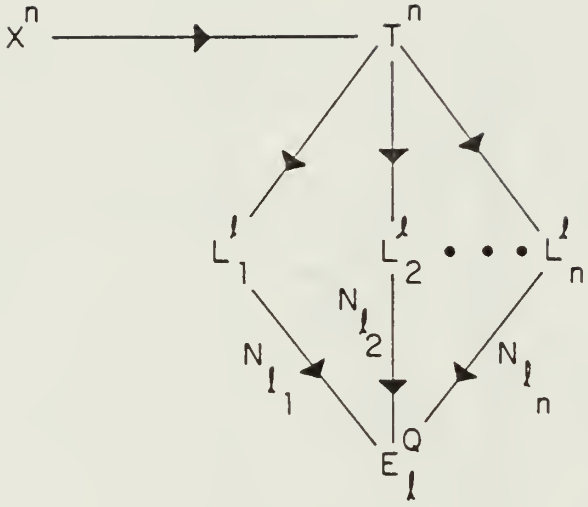


Figure 3.7 Case IV: Output Feedback ($m = n$, $l < n$) Lattice Diagram.

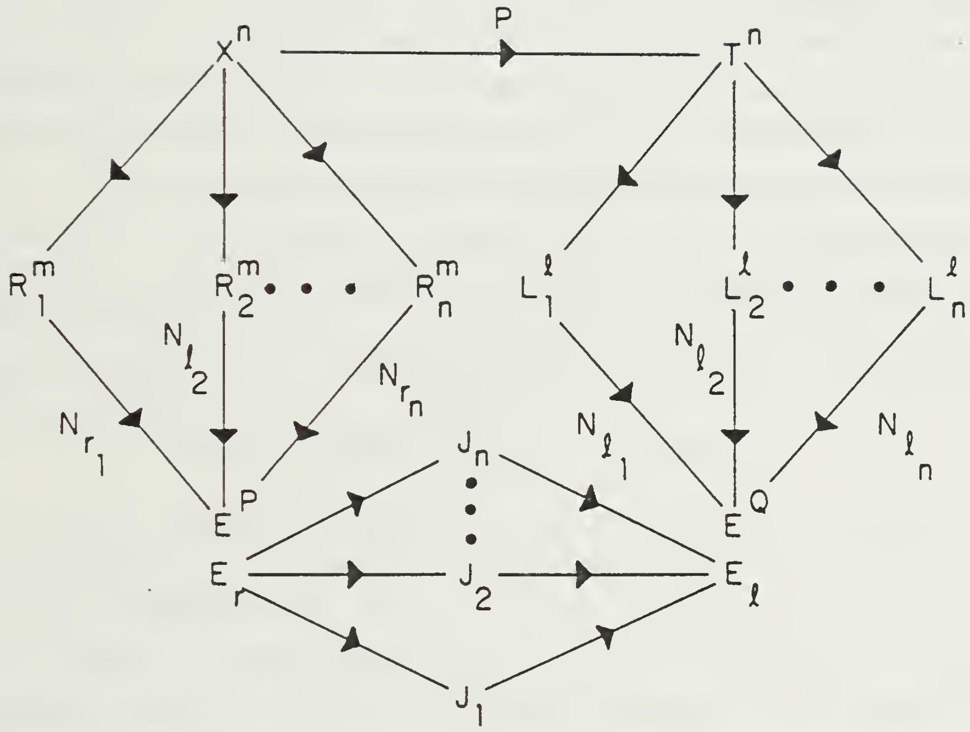


Figure 3.8 Case V: Output Feedback ($m < n, l < n, m = l$) Lattice Diagram.

IV. NUMERICAL OPTIMIZATION

A. THE AUTOMATIC DESIGN SYNTHESIS (ADS) PROGRAM

The use of numerical optimization techniques as an engineering design tool has matured with the advent of higher speed digital computers. In tandem with this growth has come the development of algorithms and design strategies [Ref. 32] available for a wide variety of particular design tasks. Several numerical optimization techniques and strategies are resident [Ref. 33] in compiled Fortran code for use at the Naval Postgraduate School. In effect, the Automatic Design Synthesis code can be used as a *black box* optimizer for a wide spectrum of disciplines. It is not the purpose here to discuss in detail all the numerical algorithms available through the ADS code, but only to describe the general design philosophy of the optimizer and to overview the particular strategies employed during the course of the thesis research.

A general purpose code, ADS contains algorithms which are able to solve the following general n-variable constrained minimization problem.

Minimize: $F(X)$

Subject to:

$$g_j(X) \le 0 \, , \; j = 1,m \quad \text{inequality constraints}$$

$$h_k(X) = 0 \, , \; k = 1,l \quad \text{equality constraints}$$

$$x_i^l \le x_i \le x_i^u \, , \; i = 1,n \quad \text{side constraints}$$

where $X = \{x_1 \;\; x_2 \; \ldots \; x_n\}^T$ is a design variable vector.

Convergence to a minimum value of the function $F(X)$ without violation of the constraint criterion in accomplished via the following iterative scheme,

$$X^{q+1} = X^q + \alpha_q^* S^{q+1} \qquad \text{(eqn 4.1)}$$

where,

q = iteration number

S^q = search direction vector

α_q^* = scalar weight applied to the search direction

The user is afforded flexibility of further tailoring the optimization scheme by being able to designate any of the following three components of the ADS optimization algorithm:

a. *Optimization strategies* such as Sequential Unconstrained Minimization Techniques (SUMT) or Augmented Lagrange Multiplier Methods which allow the user to reformulate the objective function. F(X).

b. *Search Direction* computation methods such as conjugate and variable metric first order methods which compute S^q and.

c. *One dimensional search* options such as Golden Section or polynomial interpolation methods which determine the magnitude of change in the design variable during the minimization.

As the ADS is designed to fulfill the general needs of a spectrum of engineering disciplines, the level of user sophistication beyond correct interpretation of the calling arguments is not necessary. However, as is the case of using any library type code, one must correctly formulate the objective function and clearly interpret the iterative results of the ADS code. Due to the general nature of the objective function, this last statement becomes increasingly important when little is known of the order or continuity of the function in the region of interest. Two types of constrained minimization problems were formulated for use during the research. Problem I was a constrained minimization problem with side constraints only and Problem II was a constrained minimization with inequality and side constraints. Let us discuss the strategies used for each of these two problems.

Problem I

A constrained minimization problem with side constraints may be posed as an unconstrained minimization problem. By specifying bounds on the design variables within a specified region during the course of an unconstrained minimization, one implicitly invokes the side constraints. The most efficient search direction method for Problem I was found to be the Broyden-Fletcher-Shanno-Goldfarb (BFGS) variable metric method. This is a first order method in which the gradient information is computed by ADS via finite difference calculations. Often called a quasi-Newton method, the BFGS method computes the search direction, S^q, as follows,

$$S^q = - H \nabla F(X^q) \qquad \text{(eqn 4.2)}$$

where $\nabla F(X^q)$ is the ADS computed finite difference gradient at the *qth* step and H is an iteratively computed matrix which orients the search direction. Initially $(q = 1)$ H is the identity matrix and therefore the initial search direction is in the direction of steepest descent. H is then updated after each iteration in the following way.

$$H^{q+1} = H^{q} + D \qquad \text{(eqn 4.3)}$$

where D is a symmetric matrix defined as,

$$D^{q} = (\sigma+\tau)pp^{T}/\sigma^{2} + \{H^{q}yp^{T} + p(H^{q}y)^{T}\}/\sigma \qquad \text{(eqn 4.4)}$$

The vectors p and y are defined by,

$$p = X^{q} - X^{q-1}$$

$$y = \nabla F(X^{q}) - \nabla F(X^{q-1})$$

and σ and τ are scalar vector products formed as follows.

$$\tau = y^{T} H^{q} y$$

The one dimensional search routine used was the Golden Section method. As this method is a one-variable algorithm, the single *"variable"* for our purposes is the vector, x_i or t_i. The advantages of the method include no requirement for F(X) to have continuous derivatives and in addition the Golden Section has a known rate of convergence. Among the disadvantages which impacts our purposes is that the Golden Section assumes the objective function to be unimodal, or to have one minimum in the region of search. If one has an objective function which for example has several local minima, several solutions may exist depending on the bounds set on the design variables. As will be shown subsequently, the minimization of the null vectors, ε_{ri} and ε_{li} was performed using the Problem I formulation.

Problem II

If one recasts the inequality constraint as a penalty function, P(X), where,

$$P(X) = \sum\{\max(0.g_j(X))\}^{2} \quad , j = 1,n \qquad \text{(eqn 4.5)}$$

then one may also pose Problem II as an unconstrained minimization problem in the following way [Ref. 32]. By forming a pseudo-objective function, $\Phi(X.r_p)$,

$$\Phi(X,r_p) = F(X) + r_p P(X) \qquad \text{(eqn 4.6)}$$

and by again bounding the design variables, one now has a Sequential Unconstrained Minimization problem (SUMT) with an exterior penalty function, P(X). In equation 4.6, r_p is a scalar weighting applied to the penalty P(X).

Equation 4.6 is the formulation used in this thesis for such a minimization. The search direction used for Problem II was the Fletcher-Reeves (FR) conjugate method. Like the BFGS method, the FR is a first-order method which is an improved steepest descent algorithm. The search direction, S^q for this method is defined as,

$$S^q = -\nabla F(X^q) + \beta_q S^{q-1} \qquad \text{(eqn 4.7)}$$

where,

$$\beta_q = |\nabla F(X^q)|^2 / |\nabla F(X^{q-1})|^2 \qquad \text{(eqn 4.8)}$$

As in Problem I, the Golden Section method was used for the one-dimensional search routine. Problems I and II were the ADS minimization strategies used during this research. They will subsequently be referred to as such.

B. THE NUMERICAL OPTIMIZATION PROBLEM

The fundamental expressions for feedback gains noted in Table 1 involve eigenvector constraints written in terms of null space membership. Further, when these null space constraints are expressed numerically, they take the form of equations 3.77 and 3.78 . The numerical objective than becomes one of minimizing some measure of the right and or left null vector which satisfies such a numerical null space membership. For the right eigenvector, one might write such a numerical objective as an objective function expressed in terms of a euclidean distance measure,

$$F(X) = \|U_{b1}{}^T(A - \lambda_i I_n)x_i\|_2 = \|\varepsilon_{ri}\|_2 \qquad \text{(eqn 4.9)}$$

where $\|\varepsilon_{ri}\|_2$ is the euclidean distance between the allowable subspace defined by the desired closed loop eigenvalue, λ_i , and the desired right eigenvector, x_i . As the system is time invariant, the design freedom, or variables, would be all (unconstrained minimization) or some (constrained minimization) of the right eigenvector elements and not the open loop parameters.

1. State Feedback Controller Synthesis

For purposes of illustration, let us assume one is synthesizing a state feedback regulator where there are *n = four states* and *m = two inputs*. Further, one desires the following closed loop eigenvector, x_i, to be associated with the closed loop eigenvalue, λ_i.

$$x_i = \{x_1 \;\; 0 \;\; x_3 \;\; 0\}^T \qquad \text{(eqn 4.10)}$$

where x_1 and x_3 are unconstrained and may take on any numerical value. The constrained values, noted by the zeroes in elements two and four, are desired, for example, in order to satisfy a decoupling specification. Mathematically, this is a constrained minimization problem and may be stated in the following way,

Minimize: $F(x_i)$

where,

$$F(x_i) = \| U_{b1}{}^T (A - \lambda_i I_n)(x_i) \|_2 = \| \varepsilon_{ri} \|_2$$

Subject to:

$$x_2 = 0$$

$$x_4 = 0$$

For real λ_i, this is a four variable constrained minimization problem with two side constraints. By bounding the design space ($x_2 = x_4 = 0$), this becomes a Problem I formulation. In terms of real cost, this particular synthesis problem would entail '*n*' such Problem I minimizations before a feedback solution is computed. The side constraints may also differ from vector to vector. Note further that for systems with complex eigenvalues, the computational cost would decrease since one may employ the necessary conjugate condition for the associated complex eigenvector. At this point, the minimization problem can be executed, the resulting eigenvectors are checked for independence, and the feedback controller can be synthesised. A robust decoupling controller for the CH-47 helicopter was synthesised by this method. Results will be shown in Chapter VI.

2. Output Feedback Controller Synthesis

Let us now assume we must synthesize an output feedback controller with four states, two inputs, and *l = 3 outputs.* Computational cost is increased by the output feedback requirement since in addition to the right eigenvector constrained minimization problem, one of course must also satisfy the left eigenvector constraint. One may pose this synthesis problem in a variety of ways. The following scheme was successfully formulated during the research reported in this thesis.

a. Converge on 'n' right eigenvectors by solving 'n' Problem I minimizations.

b. Compute the following right residual sum.

$$RES_r = \sum \| \varepsilon_{ri} \|_2 \,,\ i = 1,n \qquad \text{(eqn 4.11)}$$

c. Compute the corresponding left eigenvectors, t_i, by the identity,

$$T = X^{-T} \qquad \text{(eqn 4.12)}$$

d. Compute the 'n' left null vectors, ε_{li} via equation 3.78 and compute the following left residual sum.

$$RES_l = \sum \| \varepsilon_{li} \|_2 \,,\ i = 1,n \qquad \text{(eqn 4.13)}$$

e. Convergence criteria is now tested as the sum of the right and left residual sums,

$$R(X,T) = \sum \{ \| \varepsilon_{ri} \|_2 + \| \varepsilon_{li} \|_2 \} \leq \rho \qquad \text{(eqn 4.14)}$$

Since the right residual sum is satisfied initially, the left residual sum will likely be out of tolerance. The design freedom here is to allow the right eigenvectors to vary in order to satisfy the left residual. Convergence is then obtained by returning to Step B.2.b. subsequent to allowing the right eigenvector elements to change so as to minimize the left residual sum. In this way R(X,T) will be at or below a specified tolerance ρ. This method is termed the *indirect method* since the left eigenvector constraints are being satisfied by allowing the corresponding right eigenvectors to change their orientation.

One may state the output feedback constrained minimization via the *indirect* method as a Problem I formulation as follows.

$$\text{Minimize: } R(X,T)$$

where,

$$R(X,T) = \sum\{W_r\|\varepsilon_{ri}\|_2 + W_l\|\varepsilon_{li}\|_2\}$$

Subject to:

$$x_{lower} \leq x_{ij} \leq x_{upper}$$

where,

X = Right Eigenvector Matrix

T = Left Eigenvector Matrix

W_r, W_l = Scalar weighting applied to the residual sums

The *indirect* method for the output feedback case can become computationally inefficient for systems of high order. For example, a tenth order system with complex eigenstructure will contain twenty design variables per closed loop eigenvalue and require a tenth order complex matrix inversion during each execution of step B.2.c. Hence for large order systems, the output feedback synthesis scheme is modified in the following way. Steps B.2.a. - B.2.d. remain the same however step B.2.e. becomes a left eigenvector minimization problem in the form of Problem I.

$$\text{Minimize: } G(t_i)$$

where,

$$G(t_i) = \|U_{ct1}{}^T(A^T - \lambda_i I_n)(t_i)\|_2$$

Subject to:

$$t_{lower} \leq t_{ij} \leq t_{upper}$$

Upon convergence to an allowable t_i, the right null vectors are then computed to check if the right eigenvectors have been oriented out of the allowable right eigenspace. After several such iterations, one begins to adaptively learn the behaviour of the system and through interactive computer execution, one will converge on a final solution. This technique was successfully performed on an F/A-18 dynamic model.

Note that an advantage of the indirect method is that one can exercise control on the bounds of the design variables through the scalar weightings W_r and W_l and therefore exercise some limits on the orientation matching.

3. A Robust State Feedback Controller Synthesis

For multi-variable control systems, the minimum singular value of the return difference matrix [Ref. 34] can be used as a robustness criterion.

$$\sigma_m(I_m + FG) \geq \alpha \qquad \text{(eqn 4.15)}$$

where $G = C(sI - A)^{-1}B$. If $\alpha = 1$, then the feedback gains, F, in equation 4.15 may be considered *quasi-optimal* since $\sigma_m \geq 1$ for optimal state feedback controllers when the input weighting matrix is the identity matrix [Ref. 35]. By subtracting α from both sides of equation 4.15, one formulates a new objective function K(X), where,

$$K(X) = \sigma_m(I_m + FG) - \alpha \qquad \text{(eqn 4.16)}$$

Substituting the state feedback expression for F from Table I yields

$$K(X) = \sigma_m\{I_m + Z_b^{-1}U_{bo}^T(X\Delta X^{-1} - A)G\} - \alpha \qquad \text{(eqn 4.17)}$$

Successful minimization of K(X) with the constraints (g_j) that each desired right eigenvector be a member of the allowable subspace invoked by the desired closed loop eigenvalue will result in a robust quasi-optimal state feedback controller. The uniqueness of this formulation is that one can numerically synthesize an optimal state feedback controller with eigenstructure specification. This technique successfully designed a quasi-optimal decoupling controller for the CH-47 and will also be presented in Chapter VI.

4. Feedback Controller Synthesis via Eigenvalue Shifting

In B.1 and B.2 above, the design variables are the eigenvector elements as the closed loop eigenvalue remains fixed. One may recast the objective function by allowing the eigenvalue, λ_i, to assume the role as the design variable while maintaining the eigenvector elements fixed. In this way, the feedback synthesis via *eigenvalue* shifting takes the form of the following minimization problem.

$$\text{Minimize } H(\lambda_i)$$

where,

$$H(\lambda_i) = \|U_{b1}{}^T(A - \lambda_i I_n)(x_i)\|_2$$

Subject to:

$$\text{real lower bound} \leq \text{Real}(\lambda_i) \leq \text{real upper bound}$$

$$\text{imag lower bound} \leq \text{Imag}(\lambda_i) \leq \text{imag upper bound}$$

This is of course a Problem I minimization with only two design variables per each of the 'n' iterations. An advantage of this method is clearly one of computational cost. A disadvantage is that one might sacrifice modal damping and natural frequency at the expense of maintaining a desired eigenvector structure.

This concludes the framework for eigenstructure analysis and synthesis via specific numerical optimization strategies. Let us now turn to a discusssion of the algorithm which formulates the theory presented in Chapters III and IV.

V. EIGENS: AN INTERACTIVE DESIGN ALGORITHM

A. DESIGN PHILOSOPHY

The EIGENS program is an interactive Fortran algorithm designed for complex eigenstructure synthesis of systems up to tenth order. The tenth order limitation is easily modified for systems of higher order by increasing the array sizes for all matrices in the program. Throughout the development of the program. interactive coding was used extensively not only for on line debugging but also due to the iterative nature of numerical optimization. In addition, by the very nature of eigenstructure assignment. user familiarity with the system structure and subsequent interaction is a necessary ingredient for a converged solution of feedback gains.

In that the program developed as a research-learning program over the period of the thesis research, some of the earlier synthesis routines such as the *Moore-Matrix* and *Moore-Algebraic* solutions, were never used for the results presented in this thesis. They were written early on to gain an understanding of the classical results presented by Moore [Ref. 10] and were never exercised subsequent to coding the theory presented in Chapter III. All of the results presented in this thesis were computed based on the theory of Chapter III.

The basic input data beyond the necessary open loop parameters include a desired closed loop eigenstructure. User verification of this data is by means of visual display of the data. A verification data file is also written for future reference by the user. Controllability checks of the open loop eigenvalues are then displayed to the user. Beyond these checks, the program relys on the user for guidance during the numerical synthesis portion. Some examples of the guidance needed are:

1. Upper and lower bounds on the arbitrary and specified right eigenvector elements or eigenvalues (Problem I minimization),
2. Acceptance of the closed loop design based on the null space residuals.
3. Number of iterations and values for ρ (Problem II minimization).

Thus the user becomes a required feedback loop in the synthesis process. This is the basic philosophy of the EIGENS code. The user becomes more familiar with the system as time progresses and subsequently becomes expert in the final structure of the closed loop design. A description of the major routines of EIGENS follows.

B. PROGRAM DESCRIPTION

As alluded to in previous section, there are two basic feedback gain solution procedures coded in the program. The *Moore-Matrix/Moore-Algebraic* solutions are limited to real systems where the number of outputs are greater than the number of inputs ($l > m$). The *Kautsky* solution entails the theory presented in Chapter III and is only limited by the tenth order requirement. Therefore, during the course of the subsequent discussion, when reference is made to the *Moore* solution, it will be brief.

A note with regard to the *Moore* solutions must be made. For the solutions, EIGENS requires the desired closed loop right eigenvectors to be input in the following form.

$$v_i = Cx_i \qquad \text{(eqn 5.1)}$$

where C is the output matrix and x_i is the desired closed loop right eigenvector. (Presently v_i must be input irrespective of using the Moore or Kautsky solutions). Upon completion of the *Moore* routines, the subsequent closed loop eigenvectors are then checked for linear independence. If the vectors are not independent, the user is queried as to new values of v_i in equation 5.1 for reentry into the *Moore* routines. Upon converging on an independent set of right eigenvectors, feedback gains, F, are then computed. The reader is referred to [Ref. 10] for the details of this procedure.

The EIGENS Fortran listing is shown as Appendix C. Figure 5.1 depicts the general flowchart for the EIGENS code and the reader is referred to Appendix C for detailed analysis. In the discussion which follows, only the main program and those subroutines which perform the feedback gain computations are described. Auxiliary subroutines which perform such computations as complex singular value decomposition (CSVD), complex matrix multiplication (CMAMTL), etc., will not be discussed in detail. These auxiliary programs are masked to the user and are only required for the execution of EIGENS. A description of the flowchart blocks is discussed below. Notation for the number of states, inputs, and outputs is N, M, and L respectively.

1. Upon completion of initializations, the data is read from File 01 as follows. (The format statement numbers are designated by 'xx' for ease of description and the read format is shown adjacent to each read statement statement).

```
READ(1,xx)TITLE (20A4)
READ(1,xx)N,M,L,IFEED (4I2)
```

```
          IFEED = 1 (State Feedback)
          IFEED = 2 (Output Feedback)
          IFEED = 3 (Output Feedback w/Feedfwd)
READ(1,xx)((A(I,J),J=1,N),I=1,N) (6F12.5)
READ(1,xx)((B(I,J),J=1,M),I=1,N) (6F12.5)
READ(1,xx)((C(I,J),J=1,N),I=1,L) (6F12.5)
If IFEED = 1 or 2 SKIP TO READ EIGD
READ(1,xx)((D(I,J),J=1,M),I=1,L) (6F12.5)
READ(1,xx)EIGD(I)  (Desired C-Loop $\lambda_i$) (2F12.5)
READ(1,xx)E(J,I)   (Desired $Cx_i$) (2F12.5)
READ(1,xx)VD(J,I)  (Desired $x_i$) (2F12.5)
```

2. The data is then displayed to the user for input verification.
3. After computing the open loop eigenvalues, the controllability of each λ_i is then computed and displayed by a controllability flag. If the controllability flag equals one, then the open loop eigenvalue can be shifted via state feedback. If the flag equals zero, then the open loop eigenvalue is invariant under state feedback. In this case the eigenvalue cannot be shifted.
4. Subsequent to the controllability computations, the user is then allowed to change the desired closed loop eigenstructure *only* if the *Moore* solutions are going to be invoked. The user must change File 01 to input a new desired closed loop eigenstructure if the *Kaustky* solution is to be used.
5. Select feedback gain solutions (Moore or Kautsky).
6. Display results.

As the *Kautsky* solution was used for the results presented here, this subroutine will be discussed in detail. As a final discussion of the *Moore* solutions, note that in Figure 5.1 these solutions are complete when the resulting right eigenvectors are linearly independent. The resulting feedback gains are then computed, the user is queried as to the necessity of a singular value analysis of the return difference matrix, and the final design results are displayed. Let us now turn to the *Kautsky* solution which is coded within the KVECT subroutine.

C. SUBROUTINE KVECT

Figure 5.2 depicts the flowchart for KVECT. Subsequent to initializations, the SVD of B is performed and the necessary block matrices are computed. The subroutine then performs 'n' Problem I minimizations to construct the allowable right eigenvector set nearest to the desired modal set. The user then has the option of minimizing $\sigma_m(I + FG)$ (Problem II minimization) for the full state feedback case only. If output feedback synthesis is required (IFEED = 2 or 3) and the rank of C is less than 'n'. KVECT then calls FEEDEF for further computations. Upon completion of feedback gain computations, the code returns to the main program for display of results and any further processing the user might desire.

D. SUBROUTINE FEEDEF

Figure 5.3 depicts the flow for the FEEDEF subroutine. Note that the user has access to the Problem II minimization process via designation of the scalar weightings W_r and W_l. Upon completion of the minimization and feedback gain calculations, the program returns to KVECT.

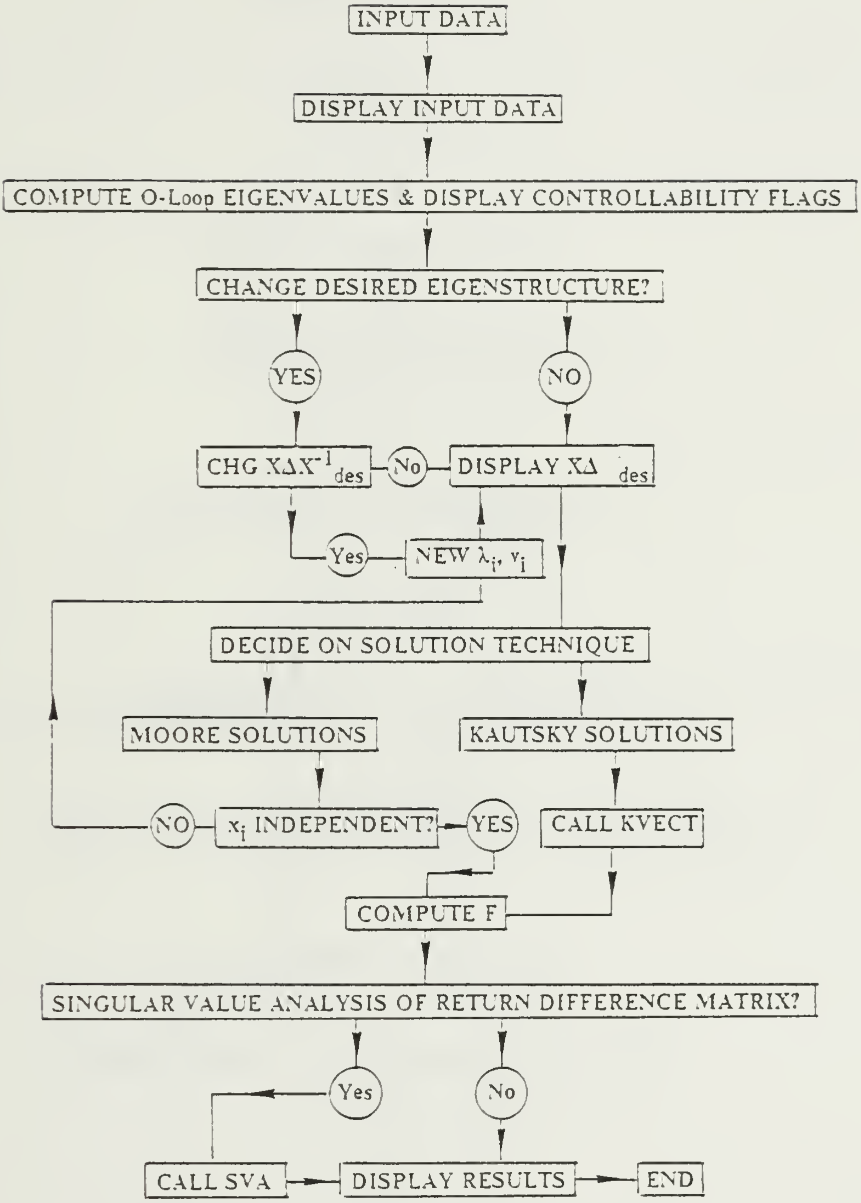


Figure 5.1 EIGENS General Flowchart.

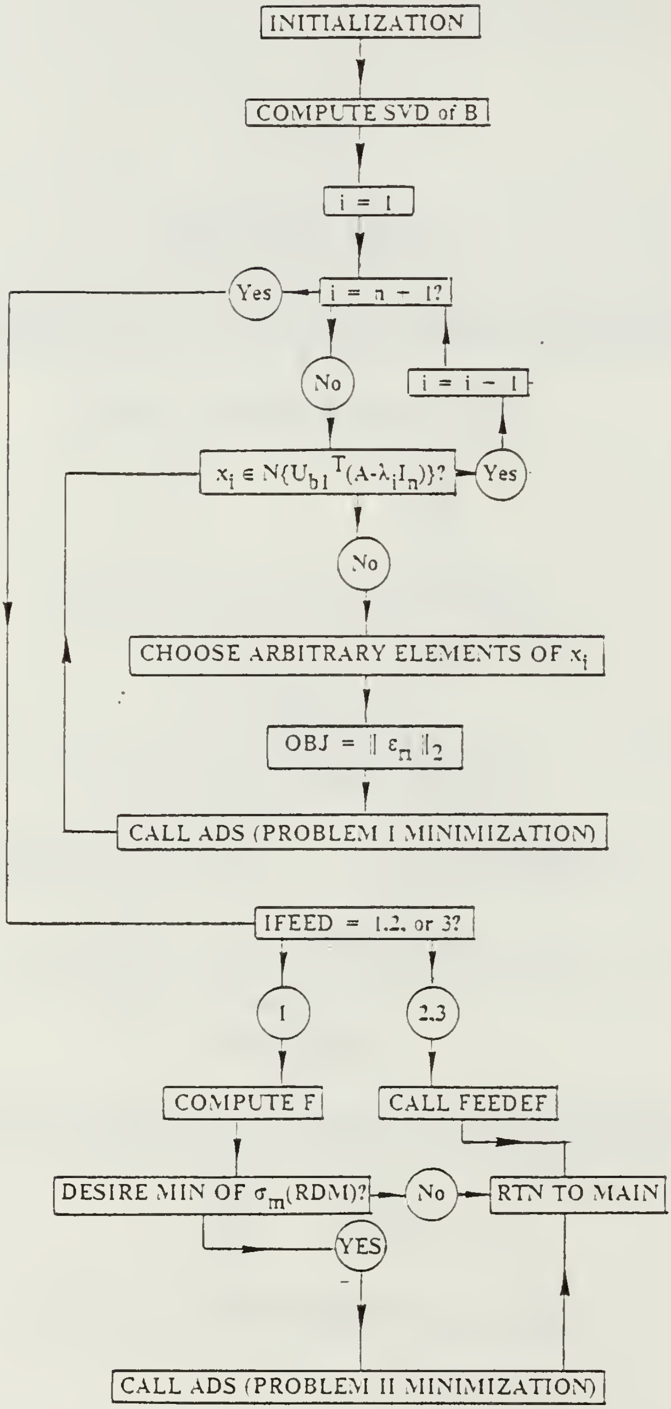

INITIALIZATION
COMPUTE SVD of B
i = 1
i = n + 1?
Yes
No
i = i + 1
$x_i \in N\{U_{b1}{}^T(A-\lambda_i I_n)\}$?
Yes
No
CHOOSE ARBITRARY ELEMENTS OF x_i
OBJ = $\| \varepsilon_n \|_2$
CALL ADS (PROBLEM I MINIMIZATION)
IFEED = 1,2, or 3?
1
2,3
COMPUTE F
CALL FEEDEF
DESIRE MIN OF σ_m(RDM)?
No
RTN TO MAIN
YES
CALL ADS (PROBLEM II MINIMIZATION)

Figure 5.2 Subroutine KVECT.

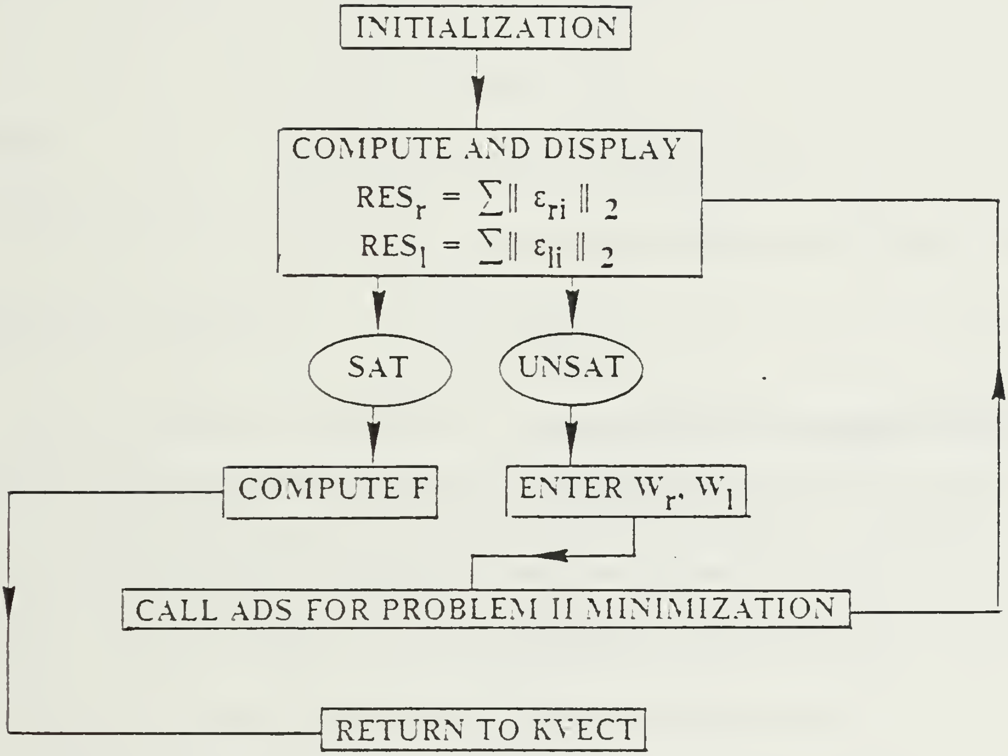

INITIALIZATION
COMPUTE AND DISPLAY
$RES_r = \sum \| \varepsilon_{ri} \|_2$
$RES_l = \sum \| \varepsilon_{li} \|_2$
SAT
UNSAT
COMPUTE F
ENTER W_r, W_l
CALL ADS FOR PROBLEM II MINIMIZATION
RETURN TO KVECT

Figure 5.3 Subroutine FEEDEF.

VI. MIMO CONTROLLER DESIGN DEMONSTRATIONS

A. A ROBUST DECOUPLING STATE FEEDBACK CONTROLLER

The lateral dynamics of the CH-47 helicopter [Ref. 36] have the following state variable representation.

$$\dot{x} = Ax + Bu \qquad \text{(eqn 6.1)}$$

$$y = Cx \qquad \text{(eqn 6.2)}$$

where the state vector is defined as,

$$x = \{v \;\; p \;\; r \;\; \varphi\}^T \qquad \text{(eqn 6.3)}$$

and the units of the vector elements are,

$x_1 = v = y$ velocity (ft/sec)

$x_2 = p =$ roll rate (rad/sec)

$x_3 = r =$ yaw rate (rad/sec)

$x_4 = \varphi =$ bank angle (rad)

The system matrices are shown in Table 2.

Lateral stability augmentation (LSA) to a roll command input, δ, is accomplished via *state* feedback with feedforward as follows,

$$u = Fx + G_2\delta \qquad \text{(eqn 6.4)}$$

where,

$$u = \{\rho_r \;\; \rho_p\}^T \qquad \text{(eqn 6.5)}$$

TABLE 2

CH-47 LATERAL DYNAMICS MODEL

A				*B*		*C*			
-2.27	-1.42	-0.15	31.99	0.12	0.95	1	0	0	0
0.01	-0.70	-0.07	00.00	0.04	-8.37	0	1	0	0
0.04	-0.05	-0.05	00.00	0.34	0.02	0	0	1	0
0.00	1.00	0.11	00.00	0.00	0.00	0	0	0	1

and,

ρ_r = yaw rate rotor deflection control

ρ_p = roll rate rotor deflection control

G_2 is the fourth column of the state feedback matrix, F.

$$G_2 = \{F(1,4) \;\; F(2,4)\}^T \qquad \text{(eqn 6.6)}$$

In that feedforward control has no influence on the resulting closed loop eigenvalues, there exist several alternatives with regard to choosing an *acceptable* form for G_2. As the system zeroes are invariant under state feedback, an *acceptable* form might be one which allows flexibility in invoking specific dynamic and static error responses to the system [Ref. 37]. No modification to G_2 was performed for this example.

The design requirements for the state feedback controller are:

a. Improve the performance and robustness of the system with respect to uncertainties at the plant input and,

b. Shift the open loop eigenvalues from,

$$\Delta_{OL} = \text{diag}\{-2.098 \;\; -1.079 \;\; 0.207 \;\; -0.050\} \qquad \text{(eqn 6.7)}$$

to the following closed loop eigenvalues,

$$\Delta_{CL} = \text{diag}\{-25.120 \;\; -12.510 \;\; -9.652 \;\; -2.125\} \qquad \text{(eqn 6.8)}$$

Improvement of the performance and robustness can be based on several measures [Ref. 38]. For the purpose of this example, increasing the minimum singular value of the return difference matrix, $\sigma_m\{I_m + FG(j\omega)\}$ for $\omega \leq 50.0$ radians was used as a robustness criterion. G is the transfer function matrix between the input u and output y, $C(j\omega I_n - A)^{-1}B$. In the discussions below, the term *MIMO robustness* is synonymous with the $\sigma_m(I_m + FG)$ defined above.

Sandell [Ref. 36] et. al. presented three different feedback gain solutions which successfully invoked the Δ_{CL} above. Two of the three lacked MIMO robustness since σ_m of the return difference matrix had values of as low as 0.3. This is a prime example of the non uniqueness of solutions for F when implementing a pole placement algorithm for MIMO systems. In other words, a particular set of feedback gains, F, will invoke a specific robustness measure while simultaneously shifting the eigenvalues. Gordon [Ref. 8] exploited this non uniqueness using numerical optimization techniques and designed robust feedback controllers which simultaneously shifted the eigenvalues and set $\sigma_m(I_m + FG)$ greater than or equal to 0.6. Another way of improving the robustness of the system is by designing an optimal LQG controller. In this way the degrees of freedom beyond pole placement take the form of an optimal control solutions. Chow [Ref. 9] designed such an optimal state feedback controller while simultaneously shifting Δ_{OL} to Δ_{CL}.

The eigenstructures of these designs designated as *Non-Robust, Gordon,* and, *Chow*, are depicted in Table 3 in orders of increasing robustness. Examination of the eigenstructures reveal an increased modal decoupling of yaw rate as the robustness improves. It is this observation which prompted the use of eigenstructure assignment to compute a feedback solution. By using the following desired eigenstructure.

$$X = \begin{matrix} x & x & x & x \\ x & x & x & x \\ 0 & 0 & 0 & x \\ x & x & x & 0 \end{matrix}$$

$$\Delta = \mathrm{diag}\{-25.120 \;\; -12.510 \;\; -9.652 \;\; -2.125\}$$

where *x* is an arbitrary design variable, and 0 is a specified value for the decoupling; and by executing the State Feedback Controller Synthesis procedure of the EIGENS code, the feedback solution denoted as EIGENS in Table 3 was obtained. Note the decoupling of the yaw rate elements and the resulting Δ_{CL}. Figure 6.1 depicts the resulting minimum singular value plots of the four designs. Note that robustness and eigenvalue placement was achieved simultaneously in a rather straightforward manner using eigenstructure assignment.

TABLE 3

CH-47 DECOUPLING STATE FEEDBACK DESIGNS

Design Feedback Gains				*Eigenstructure*			
Non-Robust Design							
				-24.80	-11.42	-10.30	-2.1
1.7	23.5	-70.6	-595.	-0.416	0.641	0.701	1.000
-0.024	2.71	-0.37	7.99	0.117	0.401	0.404	-0.008
				-1.000	*1.000*	*1.000*	*-0.019*
				0.000	-0.045	-0.050	0.005
Gordon Design							
				-24.90	-11.90	-10.70	-2.34
14.2	-3.6	-78.0	61.0	-0.415	0.233	-0.338	-1.000
0.05	2.6	-0.2	15.6	-0.122	1.000	-1.000	0.031
				-1.000	*-0.124*	*-0.095*	*-0.205*
				0.009	-0.083	0.093	-0.004
EIGENS Design							
				-25.12	-12.50	-10.50	-2.12
-8.37	-2.57	-34.64	-59.94	0.002	-0.254	-0.364	-1.000
-1.79	4.05	-5.10	28.01	-1.000	-1.000	-1.000	-0.043
				-0.004	*0.012*	*0.085*	*0.335*
				0.040	0.080	0.095	0.003
Chow Design							
				-25.30	-11.62	-10.40	-2.10
-0.65	-0.36	-34.64	-6.60	0.003	-0.290	0.409	-1.000
-0.022	4.14	4.64	30.64	-1.000	-1.000	1.000	0.008
				-0.002	*0.042*	*-0.007*	*0.020*
				0.040	0.085	-0.096	-0.005

B. A QUASI-OPTIMAL STATE FEEDBACK DECOUPLING CONTROLLER

As noted previously, Chow [Ref. 9] successfully coded an algorithm which will obtain an optimal state feedback LQG control solution which invokes a desired Δ_{CL}. Table 4 depicts the eigenstructure of one of his solutions for the CH-47 lateral stability controller. Note however that the yaw rate element associated with the fast eigenvalue, λ_4 = -2.125, exhibits some coupling. By inserting a numerical zero in place of the eigenvector elemental value of 0.1737 for x_4, and retaining the rest of the eigenstructure as inputs to the EIGENS algorithm, the Yaw Decoupled design structure was obtained. The resulting minimum singular value of the return difference matrix (I + FG) for this design, however, decreased to approximately 0.72. At this point, one has an acceptable decoupling design but it is not optimal, since $\sigma_m(I_m + FG)$ is less than 1.0. Using the yaw decoupled eigenstructure as inputs to the Robust State Feedback Controller Synthesis procedure of the *EIGENS* code, a minimization of equation 4.17 resulted in a value of α of 0.905. As α is nearly one for this solution, the feedback gains are termed Quasi-Optimal and the resulting eigenstructure is shown in Table 4. Note the yaw rate decoupling throughout the eigenvector matrix. Figure 6.2 shows the progression of the minimum singular value for each of these designs and clearly notes the increased robustness of the Quasi-Optimal solution over the Yaw Decoupled solution. Figure 6.3 depicts the resulting transient response to a 2.0 sec 0.1 radian roll pulse command for the Quasi-Optimal Yaw decoupled system. Note the essentially zero (10^{-4} rad/sec) yaw rate response.

C. A ROBUST OUTPUT FEEDBACK CONTROLLER

The state variable lateral dynamics model of the L-1011 transport augmented with rudder and aileron dynamics is shown below [Ref. 24].

$$\dot{x} = A_{aug}x + Bu \qquad \text{(eqn 6.9)}$$

$$y = Cx \qquad \text{(eqn 6.10)}$$

where the augmented state vector is defined as,

$x_1 = \rho_r$, rudder deflection (deg)

$x_2 = \rho_a$, aileron deflection (deg)

TABLE 4
CH-47 OPTIMAL STATE FEEDBACK DESIGNS

Design	*Eigenstructure*			
Chow LQG	-25.120	-12.509	-9.651	-2.125
	-0.0033 0.9992 0.0001 -0.0398	-0.2493 -0.9654 -0.0022 0.0772	-0.4650 -0.8796 -0.0405 0.0916	-0.9847 -0.0179 0.1737 -0.0006
Yaw Decoupled	-25.124	-12.510	-8.094	-2.153
	-0.0033 0.9992 0.0001 -0.0398	0.2493 0.9654 0.0022 -0.0772	0.6103 0.7862 -0.0023 -0.0971	-1.0000 0.0082 -0.0004 -0.0038
Quasi-Optimal Yaw Decoupled	-24.661	-12.511	-8.780	-2.220
	-0.0018 0.9992 -0.0043 -0.0405	-0.2492 -0.9654 -0.0022 0.0772	-0.5326 -0.8409 0.0046 0.0957	1.0000 -0.0041 0.0002 0.0018

x_3 = φ , bank angle (deg)

x_4 = r , yaw rate (deg/sec)

x_5 = p , roll rate (deg/sec)

x_6 = β , sideslip angle (deg)

x_7 = f_{wo} , washout filter state

The input vector,u ,is composed of the rudder and aileron commands, ρ_c and ρ_r .

$$u = \{\rho_c \ \ \rho_a\}^T \qquad \text{(eqn 6.11)}$$

The system matrices are shown in Table 5.

Sobel and Shapiro [Ref. 24] applied the classical results of Moore [Ref. 10] to the design of a lateral stability augmentation system (LSAS) for the L-1011 aircraft. The following design requirements set forth in [Ref. 24] were used.

TABLE 5
L-1011 LATERAL DYNAMICS MODEL

A

-20.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	-25.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	1.00	0.00	0.00
-0.744	-0.032	0.00	-0.154	-.0042	1.54	0.00
0.337	-1.120	0.00	0.249	-1.00	- 5.20	0.00
0.020	0.00	0.0386	-0.996	-.0003	-0.117	0.00
0.00	0.00	0.00	0.500	0.00	0.00	-0.500

B

20.00	0.00
0.00	25.00

Φ^{4x2}

C

0.0	0.0	0.0	1.0	0.0	0.0	-1.0
0.0	0.0	0.0	0.0	1.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	1.0	0.0
0.0	0.0	1.0	0.0	0.0	1.0	0.0

a. Shift the open loop roll and dutch roll eigenvalues to -1.5 ± 1.5j and -2.0 ± 1.0j respectively, and,

b. Decouple roll rate and bank angle from the dutch roll vectors and decouple yaw rate and sideslip angle from the roll vectors.

This design problem was executed via the Output Feedback Synthesis procedure of the EIGENS code subsequent to a model reduction. Since the first order actuator eigenvalues were an order of magnitude greater than the aircraft and washout filter poles, the faster actuators were ignored during the design synthesis. Upon computing the feedback gains, the slow system was then augmented to include the faster actuator dynamics prior to transient response analysis. This technique of eigenspace separation becomes a computational necessity when synthesisizing controllers for high order augmented linear systems. An augmented linear dynamic model of the F/A-18 tactical aircraft for example [Ref. 39] consists of fifty-five state variables.

Upon reducing the model to five states, the following desired eigenstructure was input to the EIGENS code.

$\lambda_{1,2}$	$\lambda_{3,4}$	λ_5
-1.5±1.0j	-2.0±1.0j	-0.6

$x_{1,2}$	$x_{3,4}$	x_5
0	x	x
x	0	x
0	x	x
x	0	x
x	0	x

where x again denotes an arbitrary design variable. The resulting feedback gains and eigenstructures are compared with those of Sobel and Shapiro in Table 6. The closed loop responses to an initial sideslip angle of $\beta = 1.0$ degree are shown in Figure 6.4. Figure 6.5 depicts the response using the EIGENS generated gains. Figure 6.6 compares the minimum singular value plots of the two designs. The designs have nearly identical robustness properties.

This concludes the discussions with regard to designing robust MIMO controllers. The design technique has been shown to provide flexibility towards improving an existing robust or optimal design.. It also allows the designer to exercise control over the modal content of the resulting system within the subspace constraints noted in Chapter III. It is this ***modal control*** which provided the motivation to investigate the application of eigenstructure assignment to reconfigure damaged aircraft control systems. Damage to aircraft control surfaces, wing/body profiles, or to control actuators in effect change the resulting closed loop modal eigenstructure. If one were able to regain the undamaged modal structure, then one has reconfigured the aircraft control system. Let us now turn to a discussion of such a concept.

TABLE 6

L-1011 OUTPUT FEEDBACK DESIGNS

Sobel & Shapiro Results

Feedback Gains				λ_i
-3.35	0.159	4.88	0.379	-17.05
-1.42	-2.38	6.36	-3.8	-22.01
				-1.502 ± 1.497j
				-2.001 ± .9995j
				-0.6989

Natural Freq	*Damping Ratio*
2.12	0.708
2.33	0.8946

EIGENS Results

Feedback Gains				λ_i
-3.88	0.107	6.12	0.183	-16.50
-1.29	-2.64	6.51	-4.44	-21.7
				-1.79 ± 1.55j
				-2.15 ± 1.02j
				-0.698

Natural Freq	*Damping Ratio*
2.368	0.756
2.380	0.9035

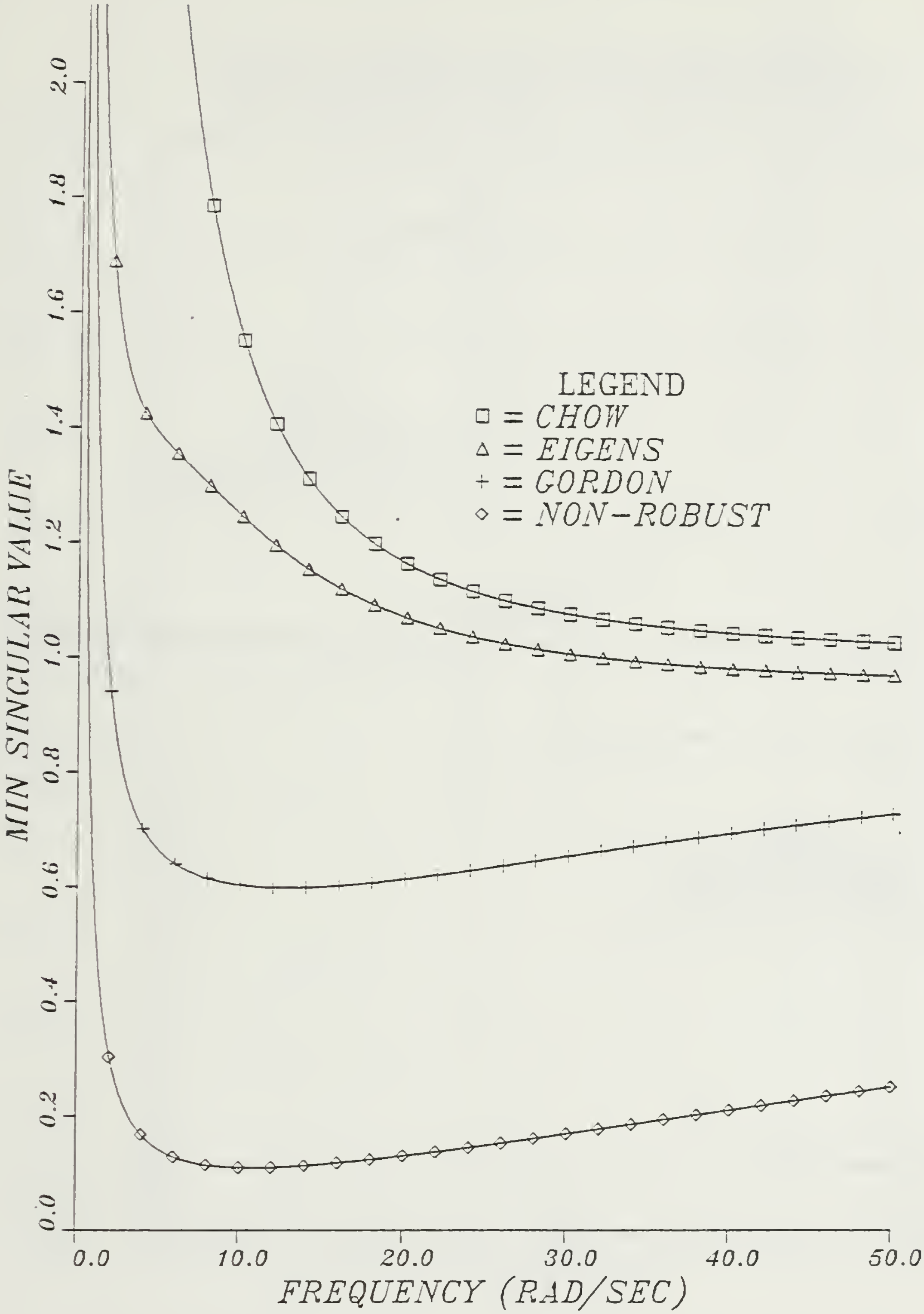


Figure 6.1 $\sigma_{min}(I + FG)$ for CH-47 Decoupled Designs.

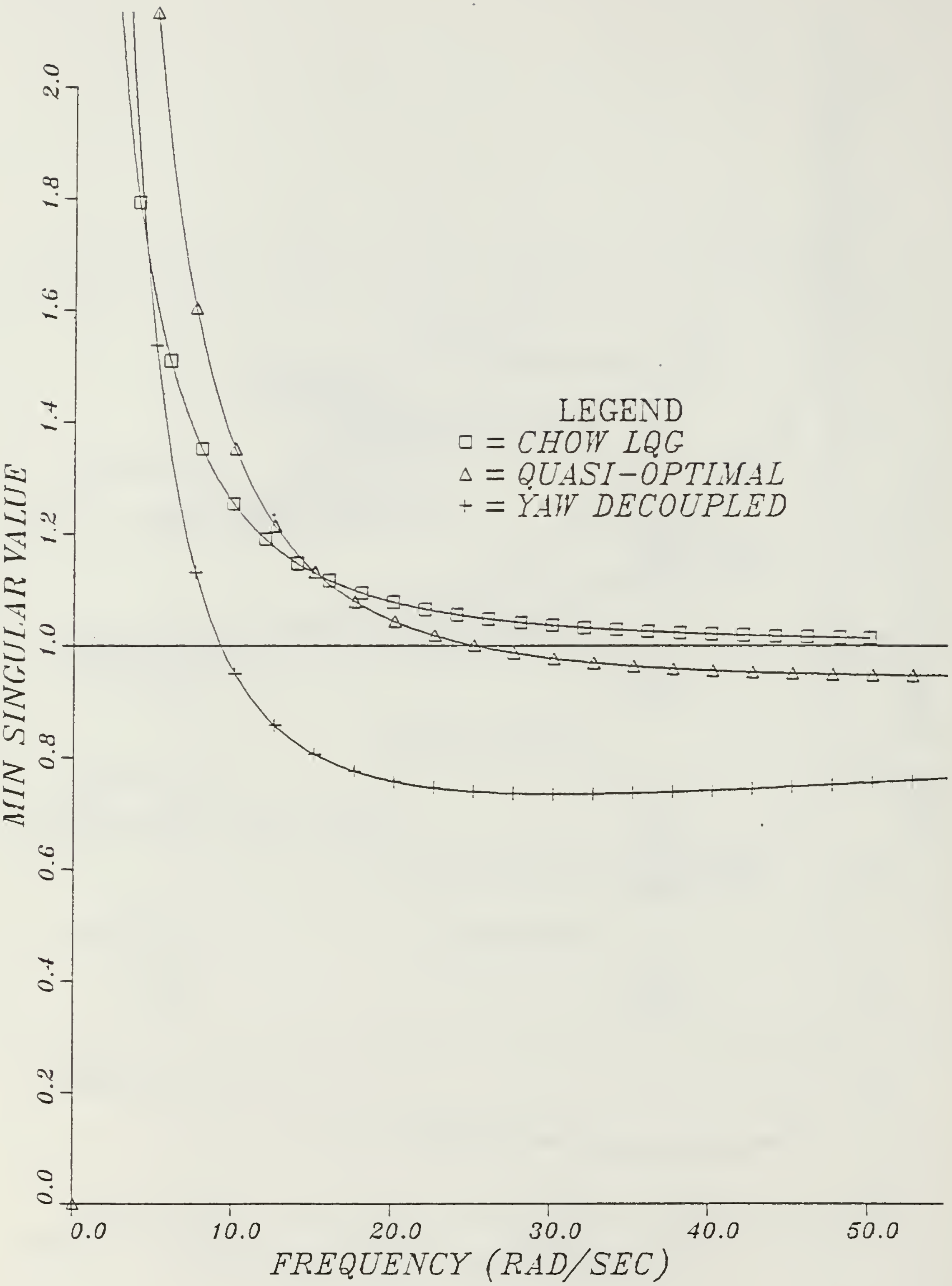


Figure 6.2 $\sigma_{\min}(I + FG)$ for CH-47 Optimal Designs.

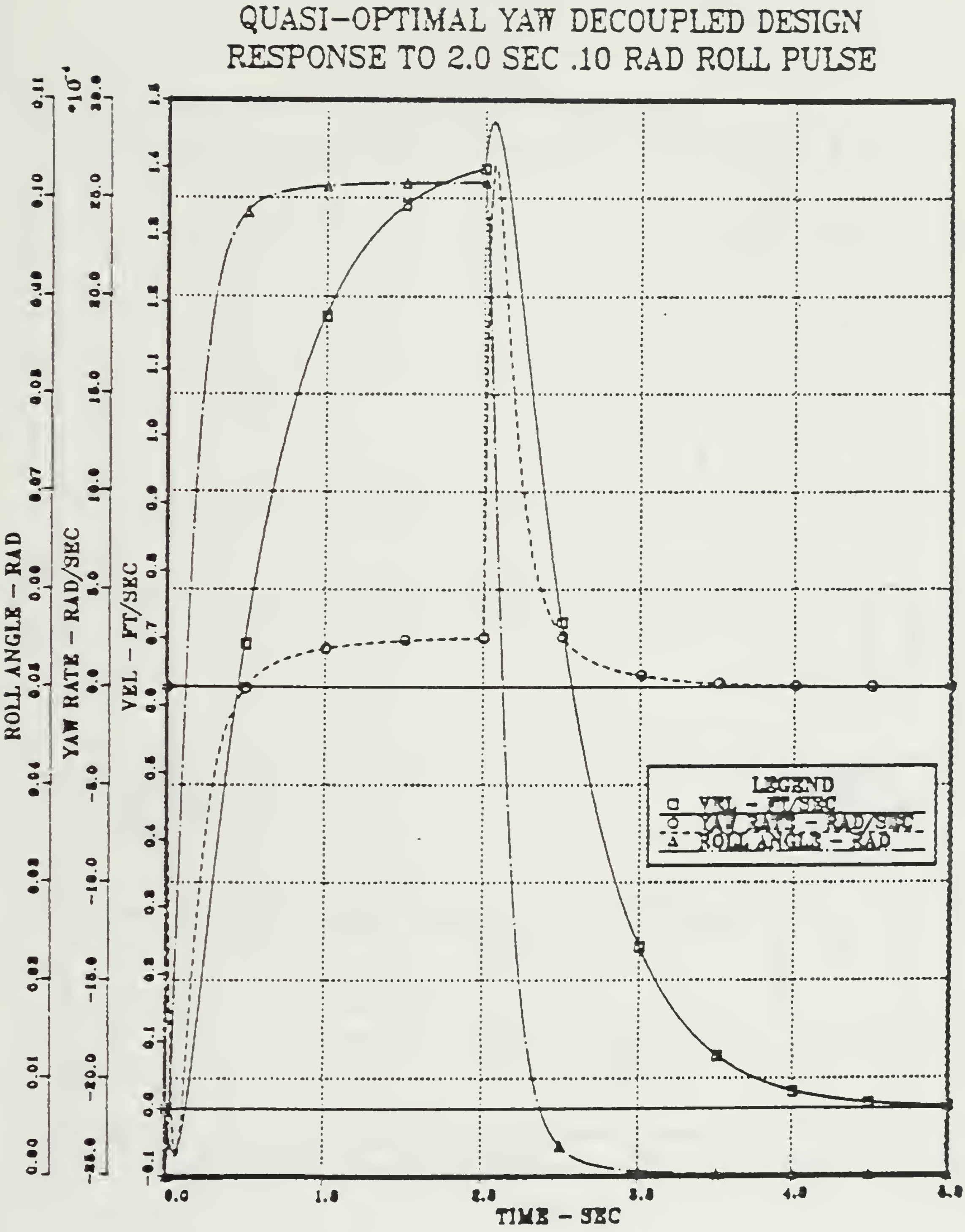


Figure 6.3 CH-47 Lateral Response for Optimal Designs.

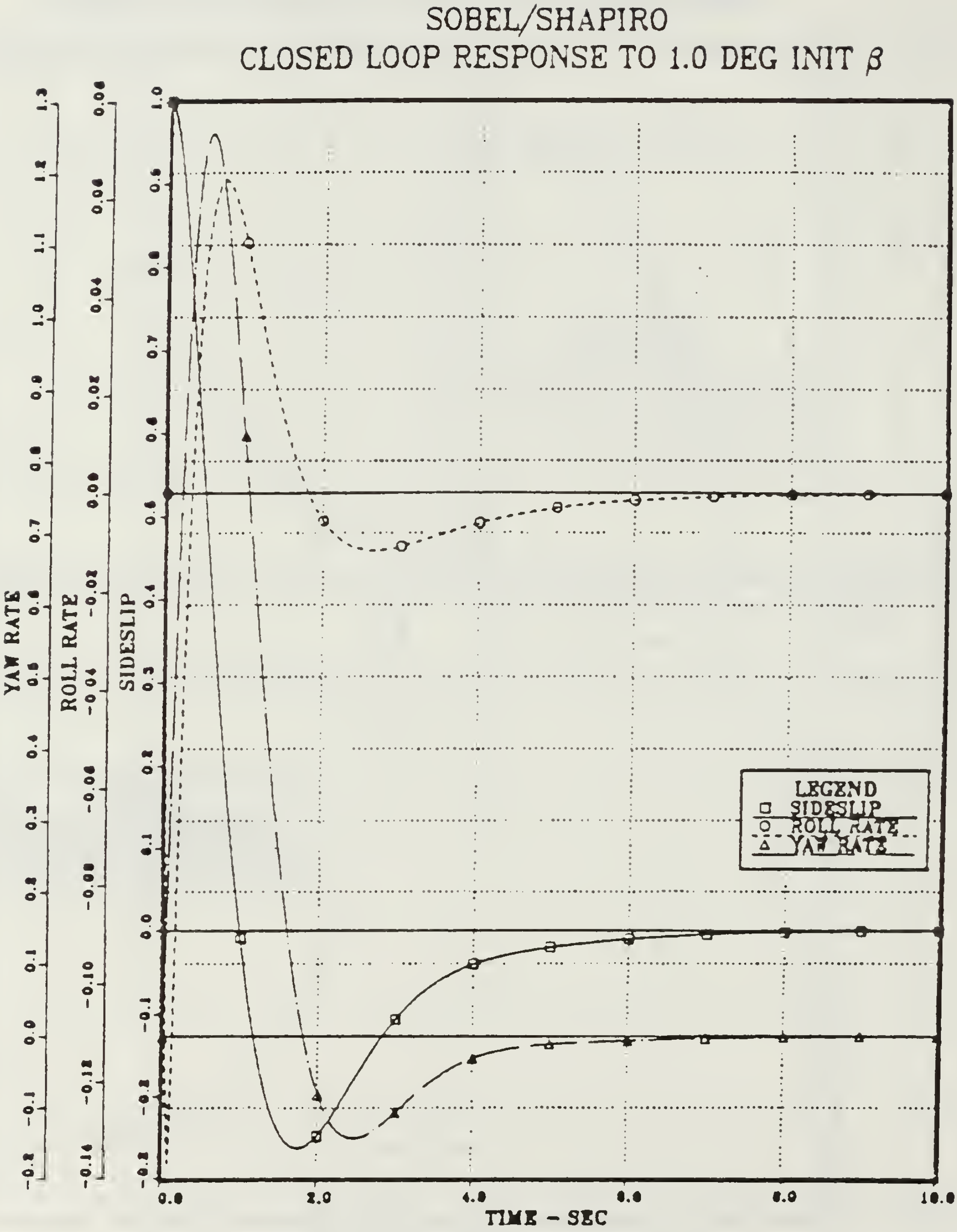


Figure 6.4 L-1011 Response to β(0) = 1.0 deg:Sobel.

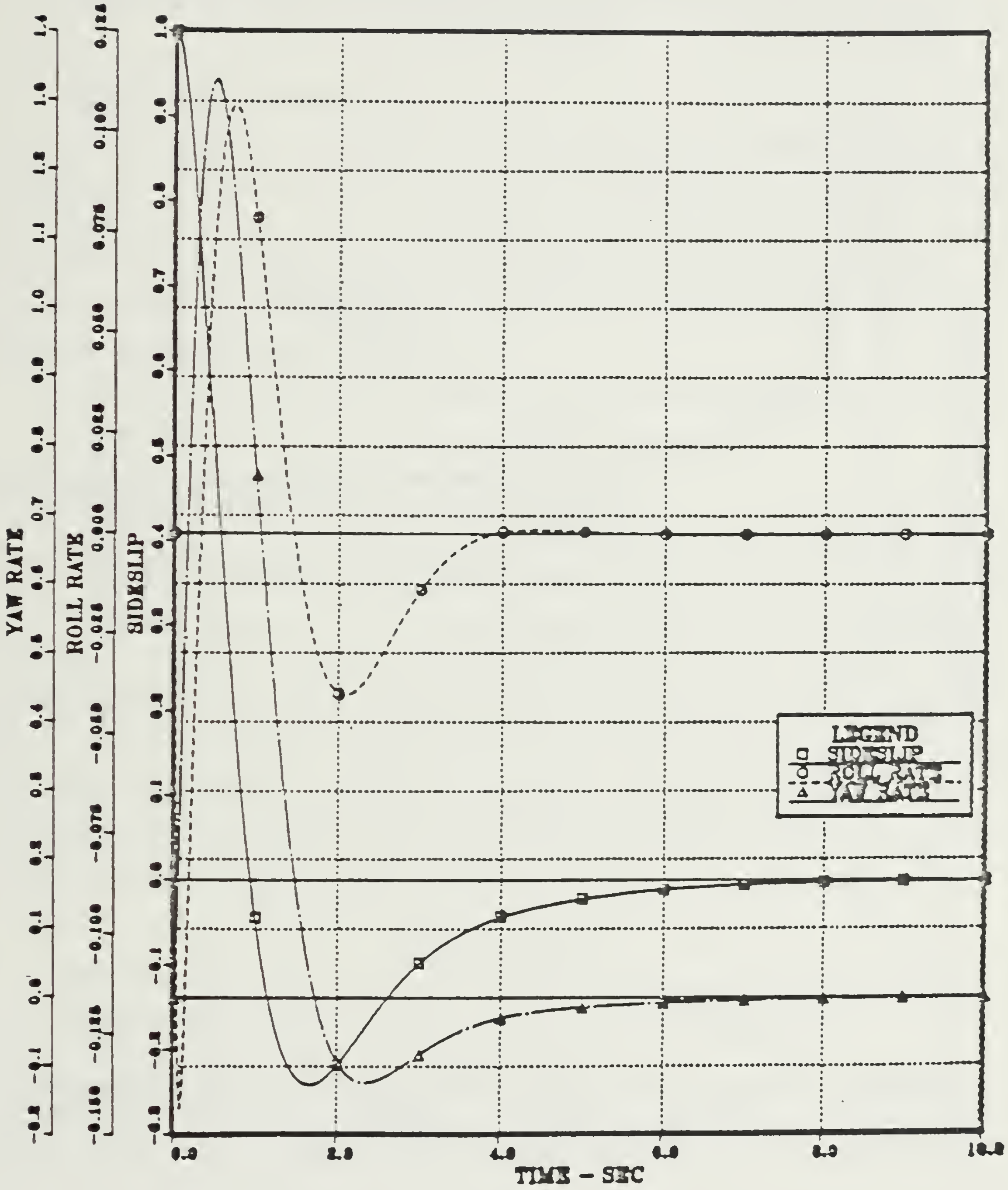


Figure 6.5 L-1011 Response to $\beta(0)$ = 1.0 deg:EIGENS.

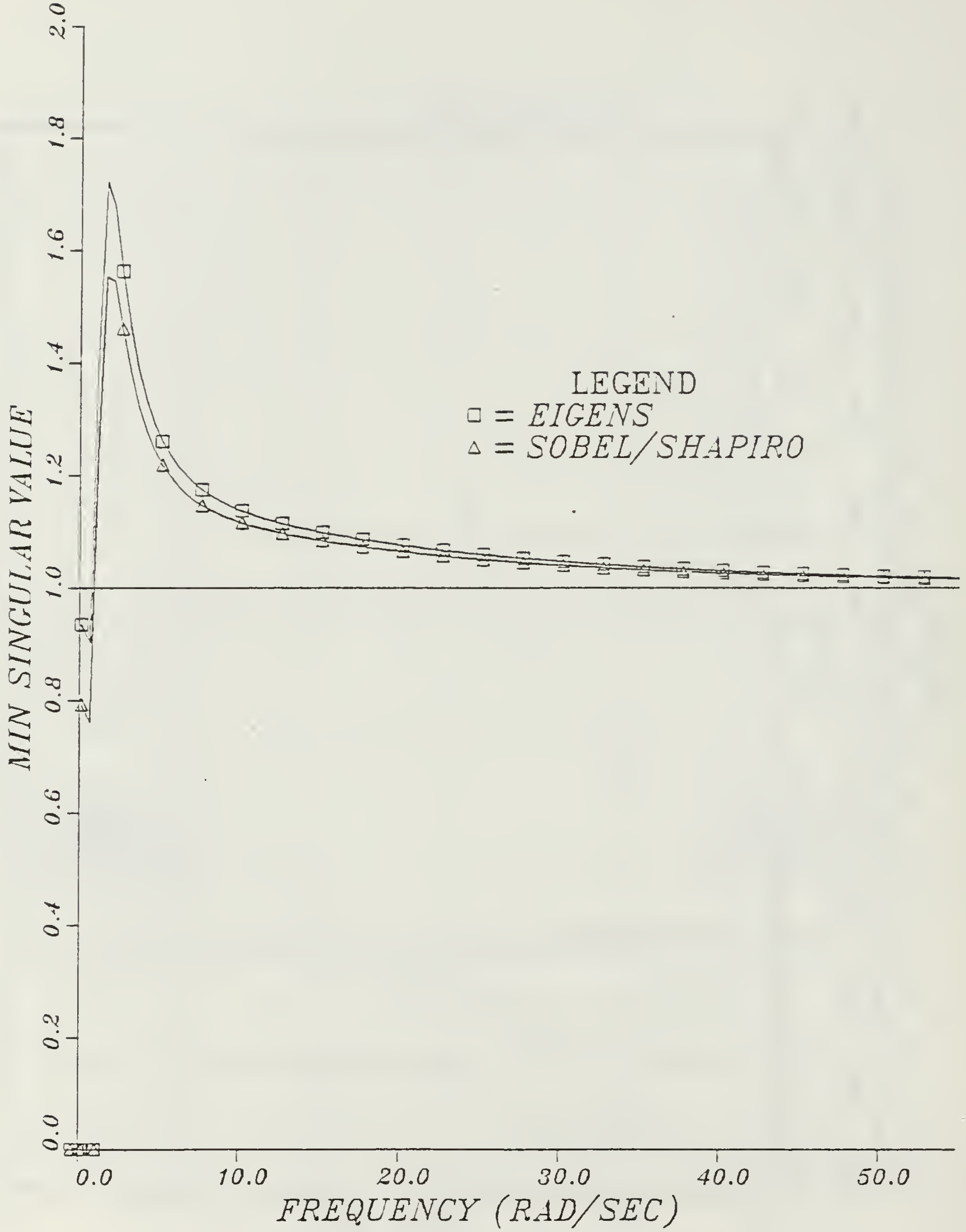


Figure 6.6 $\sigma_{min}(I + FG)$ for L-1011 Decoupled Designs.

VII. EXPLOITATION OF EIGENSTRUCTURE ASSIGNMENT TO SELF RECONFIGURING AIRCRAFT MIMO CONTROLLERS: A NEW APPROACH

A. PROBLEM STATEMENT

Prior to defining the objectives of a self - reconfiguring *MIMO* controller, one must clearly state the types of system changes and/or deficiencies which might require the flexibility of such a device. In this thesis, the design requirement of the aircraft under analysis was assumed to be the *undamaged or design dominant eigenstructure* for a specific airspeed, altitude and mission configuration. A system change or deficiency might include:

a. *Asymmetric or symmetric degradation of control surface effectiveness,*

b. *Asymmetric or symmetric control surface loss,*

c. *Single or multiple actuator degradation or failure,*

d. *Single or multiple sensor degradation or failure, or*

e. *Any aircraft damage which significantly changes the stability derivatives and therefore the modal response of the aircraft.*

The design objective of a self - reconfiguring *MIMO* controller as interpreted in this thesis is therefore to regain the undamaged modal structure subsequent to such system deficiencies via a new or reconfigured set of feedback gains. The objective is not a systems approach to reconfiguration but rather a tailored algorithm which extends the application of eigenstructure assignment. The reader is referred to [Ref. 40] as an example of what is termed a systems approach to the reconfiguration problem. A mathematical definition of such a controller might be stated in the following way.

Problem Statement

Does there exist, and if so, what are the new set of feedback gains required to regain the desired dominant eigenstructure subsequent to a system deficiency?

In terms of the overall flight control system, there are additional assumptions beside the existence of a solution which are implicit in this problem statement. The three most recognizable are:

a. *The deficiency, or damage, is assumed to be detectable in a time scale much faster than the response time of the aircraft,*

b. *Sufficient moment and force authority exist in the remaining undamaged control surfaces to overcome the deficiencies and,*

c. *Appropriate feedback loops exist between the sensed motion variable and all available control surfaces.*

Note that the objective of the reconfigured controller in this thesis is to ***regain*** the undamaged modal response with the assumption that the system deficiency has been *detected.* This is not to say however that the technique described below cannot be applied to the detection process. The concept may be equally applied to detection or observer design.

B. ALGEBRAIC SOLUTION

An algebraic solution to the problem used in this thesis is presented as follows. It was shown previously that for the output feedback case, the set of unique feedback gains, *F*, which invoke *m x l* elements of a desired right eigenvector matrix, *X*, and *max*(*m*,*l*) elements of the eigenvalue matrix, Δ, in the least square sense, are defined by,

$$F = Z_b^{-1}U_{bo}^T(X\Delta X^{-1} - A)Z_c^{-1}U_{co}^T \qquad \text{(eqn 7.1)}$$

where each right and left eigenvector must be members of null spaces defined by their respective closed loop eigenvalues, λ_i,

$$x_i \in \mathscr{L}\{U_{b1}^T(A - \lambda_i I_n)\} \qquad \text{(eqn 7.2)}$$

$$t_i \in \mathscr{L}\{U_{ct1}^T(A^T - \lambda_i I_n)\} \qquad \text{(eqn 7.3)}$$

In order to facilitate the analysis, let us rewrite equation 7.1 by substituting B_Z for $Z_b^{-1}U_{bo}^T$ and C_Z for $Z_c^{-1}U_{co}^T$,

$$F = B_Z(X\Delta X^{-1} - A)C_Z \qquad \text{(eqn 7.4)}$$

In equation 7.4 let us define the undamaged dominant modal structure, or design requirement noted above, by designating the matrix *M*, where,

$$M = X\Delta X^{-1} = BFC + A \qquad \text{(eqn 7.5)}$$

as the ***modal matrix.*** Note that system deficiencies or damage as defined previously, would result in either collective or distributional changes to the matrix triple (A,B,C). A large change in the control matrix B for example, would occur if a particular control surface was rendered ineffective by combat damage. Such a change might invoke zeroes in B and perhaps even cause rank degeneracy in the control matrix. Let us denote such changes as δA, δB, and δC. Further, if the feedback gains in 7.4 were held constant, then the modal matrix, *M*, would likewise undergo changes, δM, since if the feedback gains were to be held constant in 7.4, the following equality would arise.

$$F = (B_Z + \delta B_Z)\{M + \delta M - (A + \delta A)\}\,(C_Z + \delta C_Z) \qquad \text{(eqn 7.6)}$$

Again, cases may arise where δM could represent benign responses such as decreased pitch response due to symmetrical elevator degradation or severe changes such as an undesired roll and yaw response from pitch commands due to asymmetric elevator degradation. In order to maintain the integrity of the dominant modes of the modal matrix, *M*, constant in 7.6, and thereby regain the undamaged modal response, one could allow the feedback gains *F* to undergo a perturbation δF, to lessen the magnitude of δM. The flexibility of eigenstructure assignment and its suitability for solution by numerical optimization techniques allow one to perform such a calculation. For example, if one required the closed loop eigenvalues, Δ, to remain constant, one would allow the closed loop eigenvector matrix, X, to assume the role of design variables in an optimization routine to satisfy equation 7.2. By converging to such a solution and designating the subsequent changes to the modal matrix M as δm, where δm represents minor changes to the dominant modes, one has,

$$F + \delta F = (B_Z + \delta B_Z)\{M + \delta m - (A + \delta A)\}(C_Z + \delta C_Z) \qquad \text{(eqn 7.7)}$$

with the additional constraints that the right and left eigenvectors must now be members of the following perturbed null spaces,

$$x_i \in \mathscr{L}\{(U_{b1} + \delta U_{b1})^T(A + \delta A - \lambda_i I_n)\} \qquad \text{(eqn 7.8)}$$

$$t_i \in \mathscr{L}\{(U_{ct1} + \delta U_{ct1})^T(A^T + \delta A^T - \lambda_i I_n)\} \qquad \text{(eqn 7.9)}$$

Note that in equations 7.8 and 7.9, the changed open loop parameters are A + δA, B + δB, and C + δC and as such remain constant. If one desires a specified closed loop eigenvalue matrix, Δ, then one must allow the eigenvectors x_i and t_i to rotate until equations 7.8 and 7.9 are satisfied. If such a solution exists, then one has reconfigured the system to regain a response close to the undamaged response. Note additionally that one may apply the concept of eigenvalue shifting described in Chapter VI, Section B.4, to satisfy the null space requirements of equation 7.8 and 7.9. Equations 7.7, 7.8, and 7.9 are the key relations used in the reconfiguration algorithm presented in this thesis.

C. RECONFIGURATION ALGORITHM

The reconfiguration algorithm presented in this thesis which codes the key relations noted above can facilitate analysis of a fifty - five state variable model with up to ten inputs and eighteen inputs. It is composed of three sequentially interdependent Fortran algorithms (*RECONF, ERSPACE, ELSPACE*) which execute the reconfiguration constraints noted in equations 7.7 through 7.9 It is assumed that the user has independently computed a set of feedback gains for a particular system which invoke a satisfactory eigenstructure. The reconfiguration algorithm initializes the design analysis with the user provided data and computes the modal matrix, *M*, for further use. The algorithms were coded for execution on the Naval Postgraduate School IBM - 370 mainframe via interactive access through compatible user terminals. The reader is referred to Appendices D, E, and F for the program listings. A sequential description of a typical reconfiguration flow is described below. Acronyms to the left of the program name are designated as *read* files, those to the right are designated *write* data files.

a. *RECONF* reads the undamaged system matrices (A,B,C,F) and writes the undamaged modal eigenstructure (X,Δ).

RECONF---->----{RECONF}---->----UNDEIG

b. *RECONF* reads the user provided damaged system matrices (*RECOXX*) and undamaged modal eigenstructure (*UNDEIG*) and writes the appropriate *right eigenstructure* data (*ERSPACE*) for right null space analysis (7.8). The user has control over the order of the design space by designating the upper and lower bounds of the eigenvalues of interest. In this way, the user may synthesize the entire eigenspace by segmenting the whole space into iterative segments of design spaces or may specify a reduced order design eigenspace.

RECOXX ---->----{RECONF}---->----ERSPACE
UNDEIG

c. *ERSPACE* reads the right eigenspace data and calls the ADS program to minimize a specific normed vector distance between the undamaged right eigenvector and the perturbed (damaged) right null space (7.8). Upon convergence, the appropriate left eigenstructure data is written (*ELSPACE*) for left null space analysis (equation 7.9).

ERSPACE---->----{ERSPACE}---->----ELSPACE

d. *ELSPACE* reads the eigenstructure data and uses the ADS program to minimize the euclidean distance between the desired left eigenvector and perturbed (damaged) left null space. The user has the option of performing the minimization by either eigenvector or eigenvalue shifting. After convergence is obtained for each left eigenvector, the right null space residual is also displayed to alert the user of any conflict with the resulting right eigenvector null space membership. If such a conflict arose, the code could be modified to rewrite the ERSPACE data file for reentry into the ERSPACE algorithm. No such requirement has been noted for the model under analysis for the research reported in this thesis. This is due to the fact that for the F/A-18A linear dynamic model presented in Chapter VIII, the dimensions of E_r and E_l are 45 and 37 respectively. Therefore $d(E_r) > d(E_l)$. Upon completion of the left null space analysis, ELSPACE writes the optimized eigenstructure (OPTEIG) for use by RECONF to calculate the reconfigured gains.

ELSPACE---->----{ELSPACE}---->-----OPTEIG

e. *RECONF* reads the optimized eigenstructure and calculates the resulting feedback gains (FRECON), reconfigured eigenstructure (RECEIG) and required time response data for plotting. (*OPTXXX*) for plotting. Plotting is performed by auxiliary programs not described in this thesis.

RECOXX---->----RECONF---->-----FRECON, OPTEIG
RECEIG, OPTXXX

A sample run is presented in Appendix G.

VIII. A RECONFIGURED SOLUTION DEMONSTRATION

A. BACKGROUND

The F/A-18A was chosen as a demonstration vehicle for the solution technique posed in Chapter VII due to its digital flight control system and availability of control surfaces for reconfiguration. A current tactical aircraft in the U.S. Navy and Marine Corps inventory, its primary flight control is baselined as a *control augmentation system* which is implemented via fly-by-wire technology [Ref. 41]. The discussion will continue with a brief description of the aircraft control system and dynamic model followed by specific damage scenarios which were treated during the research reported here.

B. F/A-18A SYSTEM OVERVIEW

The F/A-18A *control augmentation* system uses feedback gain scheduling, cross axis paths such as rolling surface to rudder, and closed loop control for acceptable flying quality and aircraft stability. Flying quality guidelines are based on military specification MIL-F-8785B Level I requirements for high maneuverability aircraft [Ref. 41]. Aircraft stability margins are per MIL-F-94900 which require 6 db of gain margin for all closed loop modes. Control law computations are performed via parallel digital processing of angle of attack, normal acceleration, and dynamic/static air data to invoke the desired responses of various flight conditions. Figure 8.1 is taken from [Ref. 41] to show the flight control system from a functional level.

Figure 8.2 depicts the ten control surfaces of the aircraft. In the undamaged configuration, longitudinal control is accomplished via *symmetric* deflection of right and left stabilators and leading/trailing edge flaps. Lateral directional control is accomplished via *differential* deflection of the right and left stabilators, ailerons, leading/trailing edge flaps, and synchronous rudder deflections. Each flight control surface is driven by electro-hydraulic servoactuators with known dynamics. Rate gyros and angle of attack sensors provide the sensed motion data which is input to the digital computers for primary data processing during the flight control mode.

As mentioned previously, the primary inner loop control is accomplished via three-axis air data and angle of attack gain scheduled feedback control. Control law computations are of two categories: *Power Approach* for take off and landing and *Auto Flap Up* for all other flight modes. Only the *Auto Flap Mode* was incorporated in the dynamic model used during the thesis.

C. F/A-18A LINEARIZED DYNAMIC MODEL

The linearized F/A-18A continuous state variable model for fighter escort configuration is as follows.

$$\dot{x} = Ax + Bu \qquad \text{(eqn 8.1)}$$

$$y = Cx + Du \qquad \text{(eqn 8.2)}$$

where the state vector is composed of both the longitudinal and lateral aircraft states,

$$x = \{u \;\; w \;\; q \;\; \theta \;\; r \;\; p \;\; v \;\; \varphi\}^T \qquad \text{(eqn 8.3)}$$

and the state vector elements are,

u = forward velocity

w = vertical velocity

q = pitch rate

θ = pitch attitude

r = yaw rate

p = roll rate

v = side velocity

φ = bank angle

The input vector u is composed of the following eight inputs,

$$u = \{dst_X \;\; dle_X \;\; dte_X \;\; \rho_{st} \;\; \rho_{le} \;\; \rho_{te} \;\; \rho_a \;\; \rho_r\}^T \qquad \text{(eqn 8.4)}$$

where *d* denotes symmetric deflections and ρ denotes differential deflections of the stabilator (st), leading edge flaps (le), trailing edge flaps (te), ailerons (a), and rudders (r). The system matrices were computed from data in [Ref. 41] for a flight condition of M = 0.6 and an altitude of 10,000 feet. They are shown in Table 7.

The output vector, y, is,

$$y = \{q \;\; n_z \;\; aa \;\; r \;\; p \;\; n_y\}^T \qquad \text{(eqn 8.5)}$$

where

n_z = normal acceleration

aa = angle of attack

n_y = lateral acceleration

D. F/A-18A AUGMENTED LINEARIZED DYNAMIC MODEL

Rojek [Ref. 39] modeled the F/A-18A aircraft dynamics augmented with actuator,sensor and feedback filter dynamics for the *Auto Flap Up* flight mode in discrete form.

Figures 8.3 and 8.4 depict the linearized longitudinal and lateral-directional control law model coded in [Ref. 39]. In the figures, F denotes the gain scheduled function values and P denotes the presence of a filter. Constant gains are denoted by numerical values. Non-linear components of the full order control law model noted in [Ref. 41] were either ignored or linearized as follows.

a. Non-linear dynamics of the control laws such as position limiters, rate limiters, dead band regions, and inertial couplings were ignored.

b. Stick and rudder dynamics were ignored.

c. A constant sample rate was assumed and therefore pre-aliasing filters were not included in the model. For this thesis, an eighty hertz sampling rate was invoked.

d. Structural notch filters were ignored as structural modes were not modeled.

e. As the model was designed to simulate cruise flight conditions, faders which smooth out discontinuities during system start up and other transitional phases were ignored.

The control law model provides gain scheduling as noted in paragraph VIII.B. Lead-lag filtering is also provided to shape the feedback responses and to ensure acceptable phase and gain margins within the feedback paths. [Ref. 39] presents detailed information with regard to descretization of the continuous actuator, sensor, and control filter models shown in [Ref. 41].

[Ref. 39] formulated the discrete F/A-18A dynamic model as a fifty-five augmented state variable system. This particular state variable form modeled the discrete pilot commands as the input vector, u(k).

TABLE 7

F/A-18A SYSTEM MATRICES FOR M = 0.6, ALT = 10,000 FT.

A

-.0133	.07127	0	-32.17	0	0	0	0
-.0728	-1.140	641.92	-1.460	0	0	0	0
0	-.0127	-.9470	.00050	0	0	0	0
0	0	1.0	0	0	0	0	0
0	0	0	0	-.2450	-646.9	.02850	32.189
0	0	0	0	.00849	-.2460	.11200	0
0	0	0	0	-.0256	0.7300	-2.830	0
0	0	0	0	0	0	1.0	0

B

0	0	0	0	0	0	0	0
-129.5	19.430	-149.0	0	0	0	0	0
-15.60	-1.609	1.4990	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	-6.930	0	0	-2.915	34.909
0	0	0	-.3960	0	-.8350	-.8960	-3.260
0	0	0	11.860	0	13.060	13.140	4.4
0	0	0	0	0	0	0	0

C

0	0	1.0	0	0	0	0	0
-.0728	-1.14	-5.749	0	0	0	0	0
0	.00154	0	0	0	0	0	0
0	0	0	0	0	1.0	0	0
0	0	0	0	0	0	1.0	0
0	0	0	0	-.2450	.77720	.02850	.015

D

0	0	0	0	0	0	0	0
-129.5	19.430	-149.0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	-6.93	0	0	-2.915	34.909

$$u(k) = \{p_x(k) \;\; p_y(k) \;\; p_z(k)\}^T \qquad \text{(eqn 8.6)}$$

where

$p_x(k)$ = pitch stick command

$p_y(k)$ = lateral stick command

$p_z(k)$ = directional rudder command

Modeling in this way incorporates both the control surface deflection commands and feedback filter states within the plant matrix, A. In that the control surface deflection commands are the desired system inputs, u, and the discrete pilot commands are the desired reference commands, $\delta(k)$, the dynamic model was assembled in a different manner than [Ref. 39]. Equation 8.6 was reformulated as a reference command vector, $\delta(k)$, and the input vector, u(k), was reformulated as the ten discrete control surface inputs as follows.

$$u = \{rst \quad lst \quad rle \quad lle \quad rte \quad lte \quad ra \quad la \quad rr \quad lr\}^T \qquad \text{(eqn 8.7)}$$

where *r* and *l* denote right and left control surfaces and *st*, *le*, *te*, *a*, and *r* denote stabilator, leading edge flap, trailing edge flap, aileron, and rudder deflections respectively. Additionally, Rojek provided longitudinal(LONG) and lateral (LATD) gain matrices which modified the equations of motion to accept individual control surface deflections as systems inputs. In this way, coupling between the longitudinal aircraft modes and longitudinal control surfaces, and also the lateral modes and control surfaces, are invoked via the gain matrix transformation. For symmetric control surface degradation where for example both right and left stabilators are degraded by an equal amount, the aircraft equations of motion remain uncoupled via the structure of the gain matrices (LONG,LATD). If only the right stabilator were degraded however, cross coupling of the aircraft modes would be invoked by the gain matrix structures.

Assembling the model in this form leads to a *classical* state variable representation,

$$x(k+1) = Ax(k) + Bu(k) + G_1(k)\delta(k) \qquad \text{(eqn 8.8)}$$

$$y(k) = Cx(k) \qquad \text{(eqn 8.9)}$$

$$u(k) = Fy(k) + G_2\delta(k) \qquad \text{(eqn 8.10)}$$

where the vectors have the following linear space dimensions, The augmented output vector, $y(k)$, is composed of the sensed motion variables and the control law filter states which arise from their state variable formulation.

$$y = \{q \;\; n_Z \;\; aa \;\; r \;\; p \;\; n_y \mid x_{cl}{}^{1x12}\}^T \qquad \text{(eqn 8.12)}$$

By examining equation 8.12 and equation 8.7 one observes the structure of the feedback gain matrix, F. Note how this structure exposes the feedback loops in the system in a numerical fashion. For example, F(1,1) is the static feedback gain between the sensed pitch rate, q, and the right stabilator deflection command, rst. This physical interpretation of the numerical structure of the system is not evident in [Ref. 39] and serves as a revealing source of information for reconfiguration analysis. Note additionally that the gain matrix, F, will remain ***constant*** unless different flight conditions and/or regimes are assumed. Additionally, if one models control surface degradation via changes in the B matrix, then these constant F values will invoke undesired modal responses subsequent to pilot commands via $\delta(k)$. It is the intent of the reconfiguration technique to regain the undamaged response with the same pilot command inputs. This, of course, necessitates changing F by some quantitative amount δF. This is exactly what we desire to determine by the solution technique described in Chapter VII. Let us now turn to some specific damage scenarios for solution.

E. THE SYMMETRIC DEGRADATION PROBLEM

Several types of flight control surface deficiencies or damage classes were presented in Chapter VII ranked in orders of severity. The initial type of damage class considered in this thesis was the symmetric degradation of control surface effectiveness. Specifically, two cases of symmetric degradation of the right and left stabilators were treated by the reconfiguration technique. *Case A* decreased the control effectiveness of both stabilators by 25%and *Case B* decreased the effectiveness by 50%. Since this type of longitudinal symmetric damage does not invoke lateral coupling, the perturbed eigenstructure only involves the longitudinal dynamics. This provides a suitable test case for the initial attempt at reconfiguration.

As is the case in all eigenstructure synthesis scenarios, one must become familiar with the structure of the system prior to any design execution. In all of the damaged cases examined for this thesis, the design requirement was to regain the undamaged

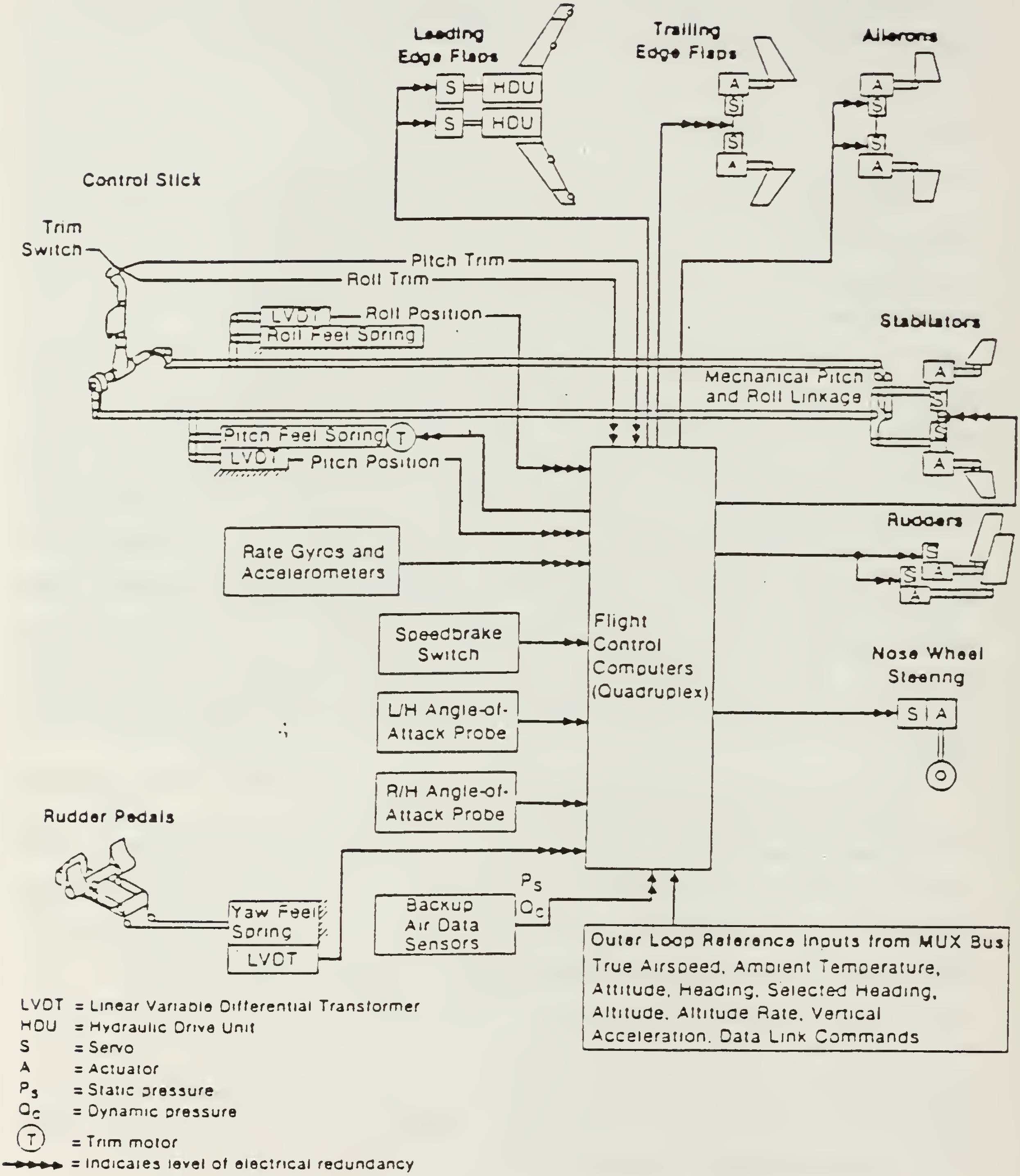
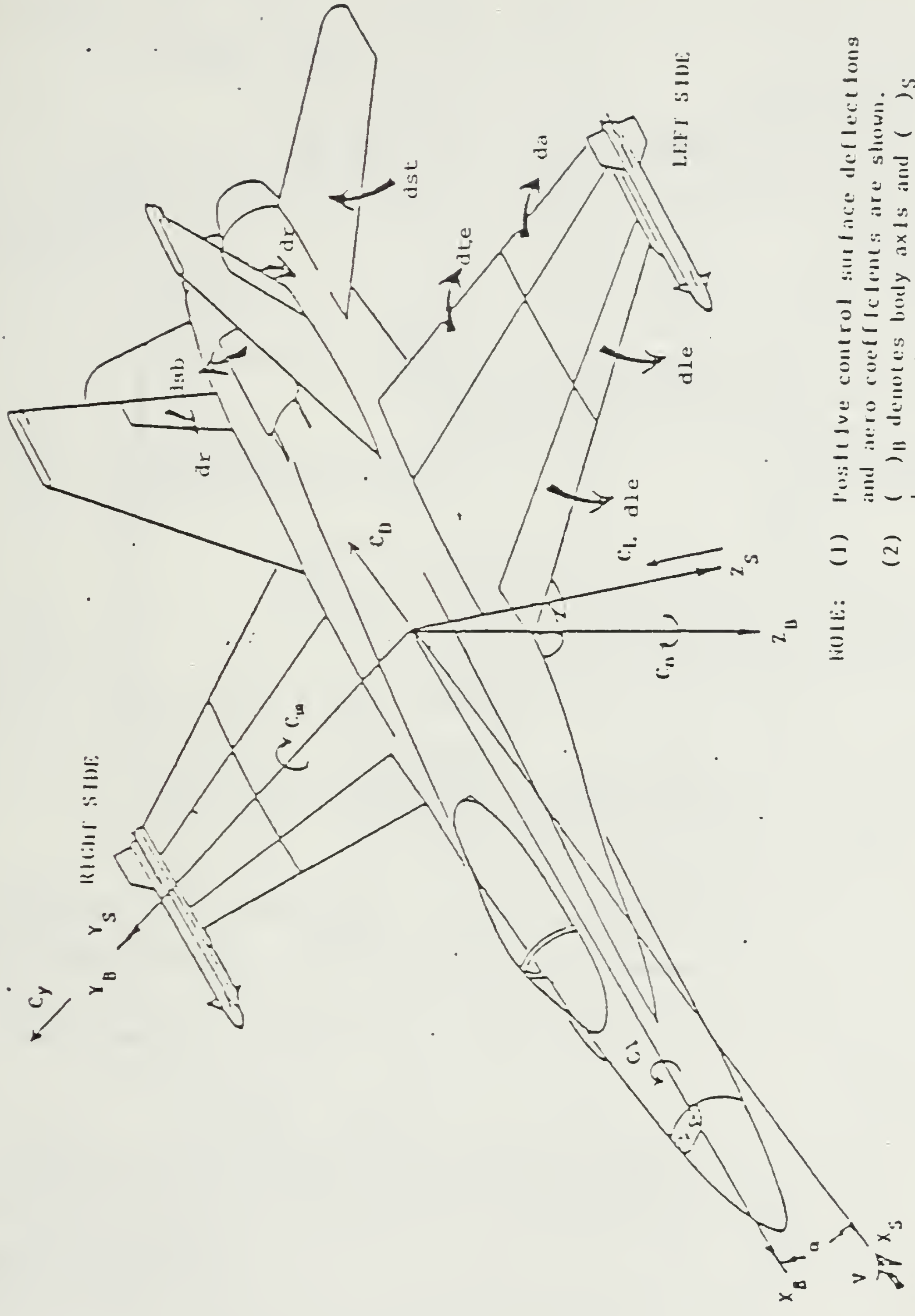

Leading Edge Flaps
S
HDU
S
HDU
Trailing Edge Flaps
A
S
S
A
Ailerons
A
S
S
A
Control Stick
Trim Switch
Pitch Trim
Roll Trim
LVDT
Roll Position
Roll Feel Spring
Stabilators
Mechanical Pitch and Roll Linkage
A
S
S
A
Pitch Feel Spring
T
LVDT
Pitch Position
Rate Gyros and Accelerometers
Rudders
S
A
S
A
Speedbrake Switch
Flight Control Computers (Quadruplex)
L/H Angle-of-Attack Probe
Nose Wheel Steering
S
A
R/H Angle-of-Attack Probe
Rudder Pedals
P_s
Q_c
Yaw Feel Spring
LVDT
Backup Air Data Sensors
Outer Loop Reference Inputs from MUX Bus
True Airspeed, Ambient Temperature, Attitude, Heading, Selected Heading, Altitude, Altitude Rate, Vertical Acceleration, Data Link Commands
LVDT = Linear Variable Differential Transformer
HDU = Hydraulic Drive Unit
S = Servo
A = Actuator
P_s = Static pressure
Q_c = Dynamic pressure
T = Trim motor
= Indicates level of electrical redundancy

Figure 8.1 F/A-18A Flight Control System.



NOTE: (1) Positive control surface deflections and aero coefficients are shown.
(2) $( )_B$ denotes body axis and $( )_S$ denotes stability axis.

Figure 8.2 F/A-18A Flight Control Surfaces.

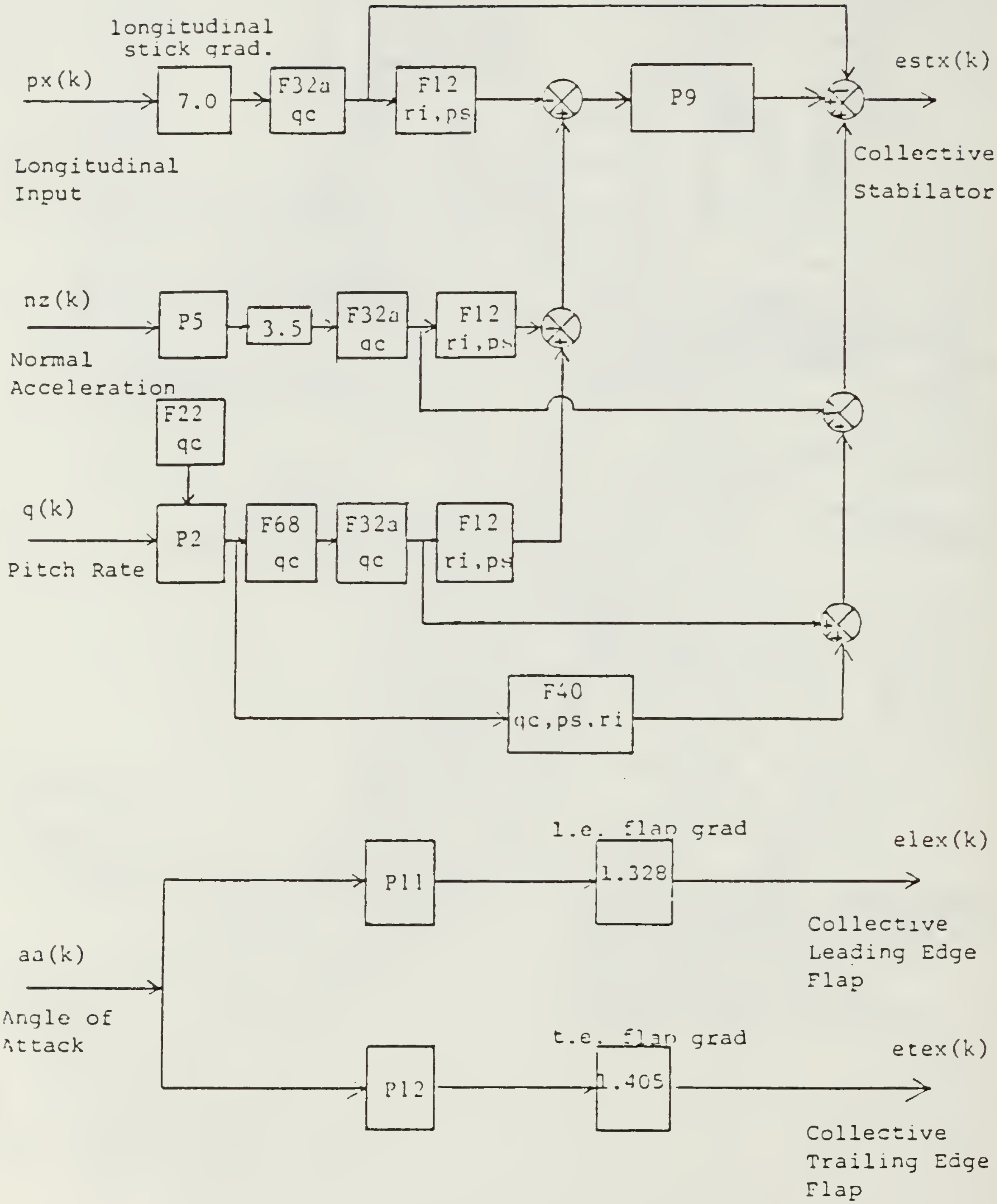


Figure 8.3 F/A-18A Longitudinal Control Model.

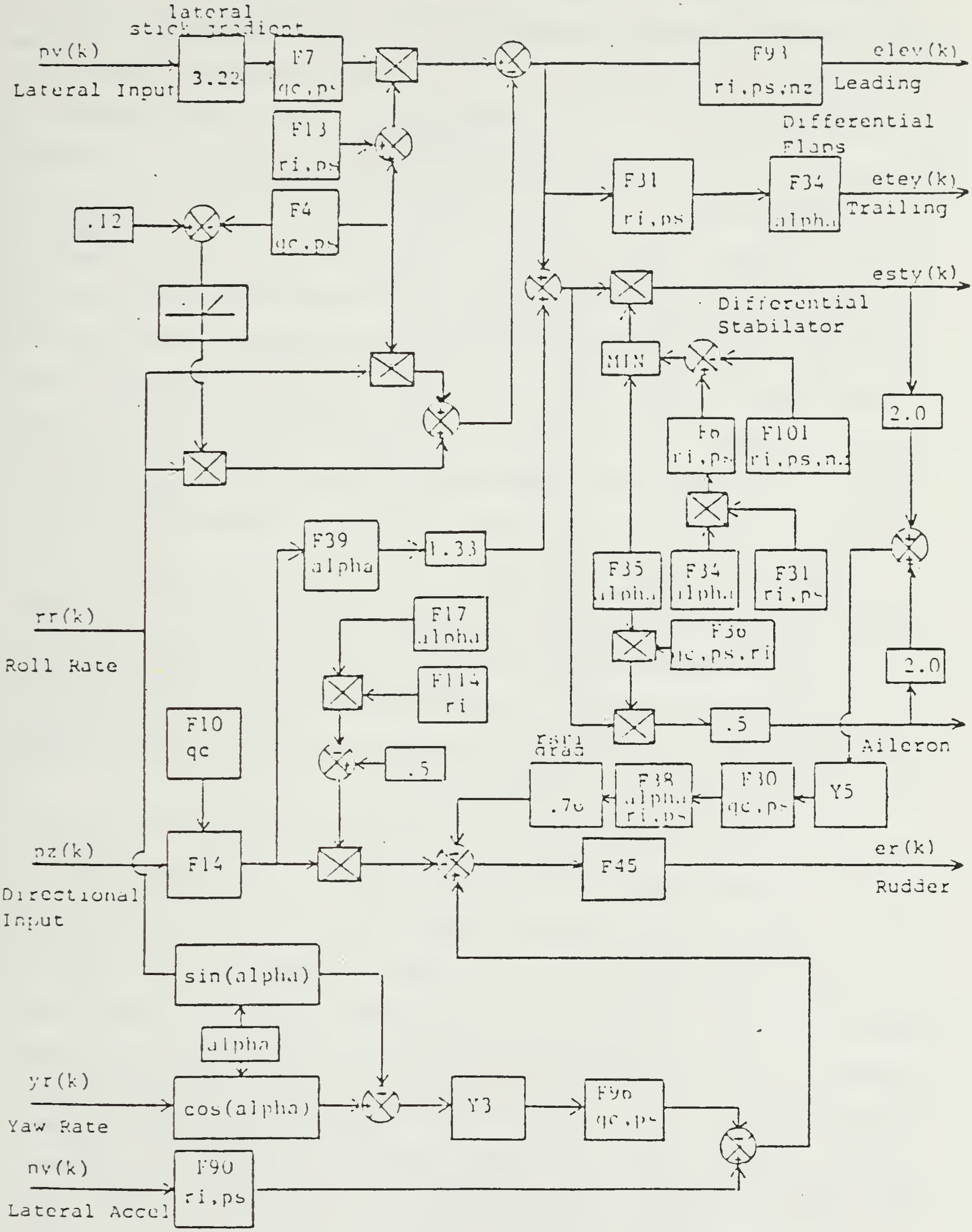


Figure 8.4 F/A-18A Lateral-Directional Control Model.

longitudinal response to a *1.0 inch, 3.0 sec pitch up stick command.* Figure 8.5 depicts the undamaged longitudinal responses to such an input. Table 8 depicts the closed loop eigenvalues of the full order system. Note that a slow eigenspace exists for a z-plane eigenvalue of greater than or equal to 0.9535. A slow design eigenspace was therefore defined as $z \geq 0.9535$ and reconfiguration synthesis for the symmetric degradation cases was performed within this eighteenth order eigenspace. Table 9 depicts the eigenvectors of the aircraft eigenspace. Tables 8 and 9 are therefore the design closed loop eigenvalues and right eigenvectors for this Case VI Output Feedback problem. In all subsequent tables, the aircraft eigenvectors are normalized with respect to the largest real magnitude of the aircraft elements. Table 10 shows the undamaged feedback gain matrix, F. Note that the function gains and filter transfer functions depicted in Figures 8.2 and 8.3 are transformed via the modeling technique noted in paragraph VIII.D into the elemental values of the feedback matrix, F, in Table 10. Each column represents the feedback gain values from the sensed variable to the respective control surface. For example the static gain value between pitch rate, q, and the right stabilator, rst, is 0.214.

Table 11 numerically depicts the euclidean and ∞-norms of the right and left null vectors, ε_{ri} and ε_{l}, of the eighteen slow modes. The residual sums, RES_r and RES_l, are the sums of the eighteen individual euclidean norms. Note that RES_r is significantly less than RES_l and that the primary modal contributors to RES_l are the short period and phugoid modes. It is interesting to note that at the outset, the *bad actors* with respect to the left null spaces are longitudinal in nature.

Upon examining Table 11, one would initially establish a *collective* numerical threshold on the null space membership criteria of 0.074838 and 3.3046 respectively. A significant unknown, however,is the magnitude of the conservative nature of the euclidean norm. A collective measure of null space residuals may indeed be too harsh a criterion on eigenvector acceptance. Additionally, as noted previously, ***eigenvector orientation*** is a significant input with regard to acceptance criteria when analyzing such a high order system. For example, a particular eigenvector may satisfy a null space requirement but have undesired modal orientation. An undesired orientation results in responses which may or may not be acceptable. The design problem therefore becomes two fold. Null space constraints and proper orientation of the right eigenvector must simultaneously occur to ensure full *performance recovery*. Performance recovery could be accomplished in several ways. One procedure would be to augment the current

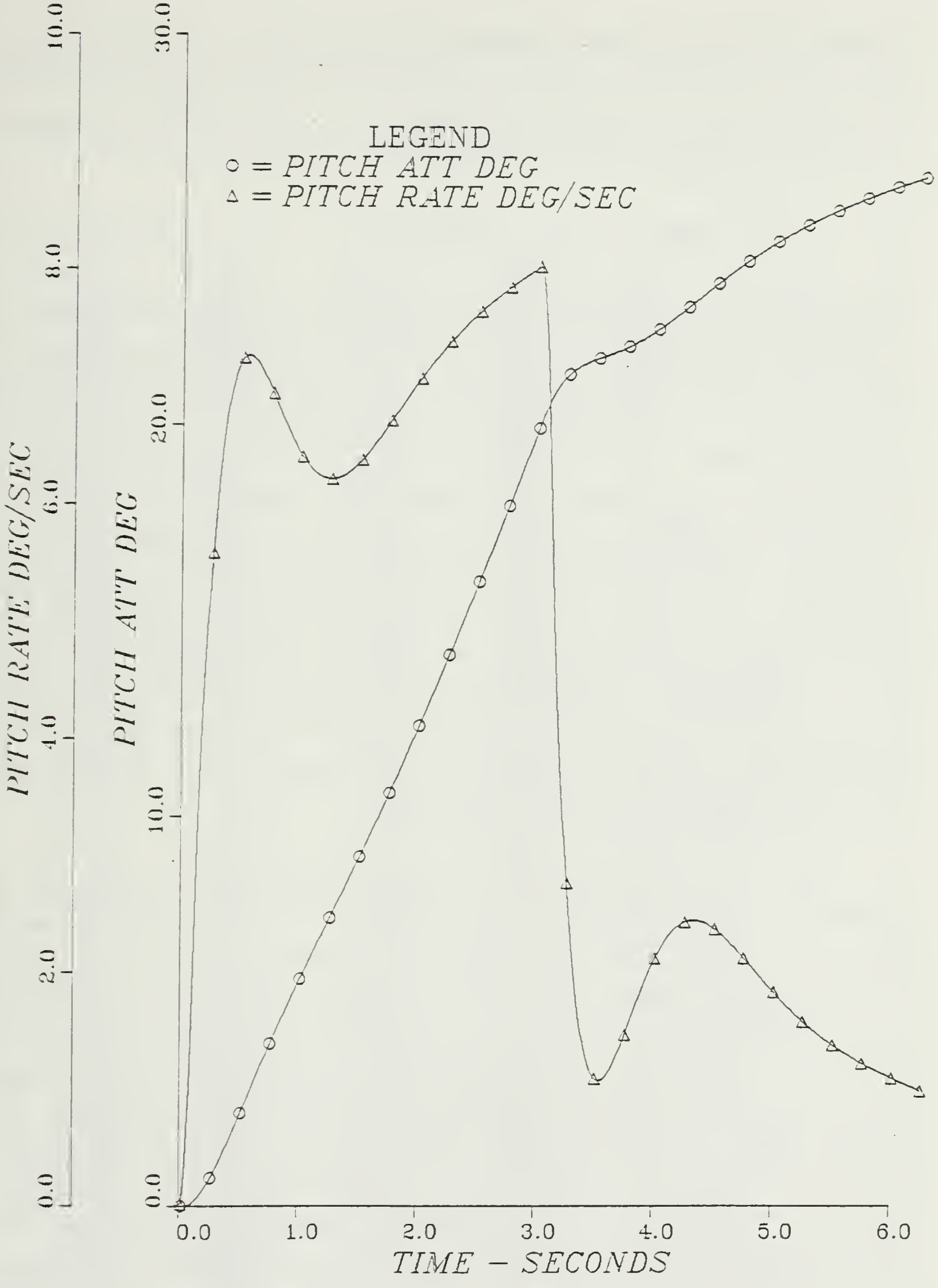


Figure 8.5 F/A-18A Undamaged Longitudinal Response.

objective function to include a performance recovery factor. One may define such a factor as a norm of the difference between the desired right eigenvector, x_i, and the right eigenvector, $x_i + \delta x_i$ which results subsequent to reorienting the left eigenvector. In this way, the augmented objective function becomes,

$$OBJ_{aug} = \|\varepsilon_{li}\|_2 + \|\delta x_i\|_2 \qquad \text{(eqn 8.13)}$$

By performing a minimization of equation 8.13, one would drive towards the undamaged performance, x_i, and the respective undamaged eigenvalue location, λ_i. Another procedure which would be significantly less expensive from a computational standpoint would involve comparing the optimized right eigenvector modes with those which are known to give satisfactory performance but possibly unsatisfactory eigenvalue placement. In other words, subsequent to minimizing a norm of the left null vector, ε_{li}, examine the reoriented right eigenvectors for structures which impede performance recovery. For those which impede recovery, drive the designed right eigenvectors to the proper orientation. This will possibly result in penalties with regard to invoking an eigenvalue location but will enhance the system performance. In this way, the designer must trade between modal performance recovery (x_i) and speed of transient response (λ_i). One must therefore observe the vectors individually with regard to both residual measurement and orientation before making prudent judgements on the acceptability of the eigenvector.

1. Case A

Symmetric degradation of the stabilators was simulated by modifying the appropriate elements of the gain matrices to reflect a 25% loss in control effectiveness. Figures 8.6 and 8.7 depict the damaged pitch attitude and pitch rate responses. Although a relatively benign perturbation to the undamaged response, this case served a valuable role in software debug of RECONF, ERSPACE,ELSPACE, and in the modifications to the [Ref. 39] code. Table 12 depicts the damaged eigenvalues for Case A. Table 13 shows the resulting perturbed aircraft eigenstructure for Case A. The primary eigenstructure perturbation for Case A is noted in the slower phugoid ($\lambda = -0.4123$) and the short period modes. For the slower phugoid mode, the eigenvalue shifted to the right by 19.4% with no significant change in the eigenvector. The perturbed short period mode became less damped and more oscillatory as the real part of the eigenvalue moved to the right by 16.7% and up the imaginary axis by 9.4%. The cumulative result is a lag in both the pitch attitude and pitch rate responses.

TABLE 8

F/A-18A UNDAMAGED C-LOOP EIGENVALUES:M = 0.6 ALT = 10K

Z-Plane Roots		*S-Plane Roots*		*Mode*
1.0000	0.0000J	0.000	0.000J	Filter
1.0000	0.0000J	0.000	0.000J	Filter
1.0000	*0.0000J*	*0.003*	*0.000J*	*Spiral*
1.0000	0.0000J	0.004	0.000J	Filter
0.9998	*0.0000J*	*-0.017*	*0.000J*	*Phugoid*
0.9936	*0.0000J*	*-0.511*	*0.000J*	*Phugoid*
0.9876	0.0000J	-1.000	0.000J	Filter
0.9856	0.0000J	-1.159	0.000J	Filter
0.9821	0.0000J	-1.443	0.000J	Filter
0.9753	0.0000J	-2.000	0.000J	Filter
0.9753	0.0000J	-2.000	0.000J	Filter
0.9753	0.0000J	-2.004	0.000J	Filter
0.9643	*0.0000J*	*-2.904*	*0.000J*	*Roll*
0.9764	*-0.0319J*	*-1.870*	*-2.615J*	*Dutch Roll*
0.9764	*0.0319J*	*-1.870*	*2.615J*	*Dutch Roll*
0.9714	*-0.0328J*	*-2.279*	*-2.702J*	*Short Period*
0.9714	*0.0328J*	*-2.279*	*2.702J*	*Short Period*
0.9535	0.0000J	-3.811	0.000J	Filter
0.8413	0.0000J	-13.829	0.000J	AoA Sensor
0.7619	0.0000J	-21.757	0.000J	Filter
0.7109	0.0000J	-27.292	0.000J	LE Flap Actuator
0.7106	0.0000J	-27.330	0.000J	LE Flap Actuator
0.6988	-0.2220J	-24.821	-24.612J	TE Flap Actuator
0.6988	0.2220J	-24.821	24.612J	TE Flap Actuator
0.7660	0.3195J	-14.909	31.616J	Stab Actuator
0.7660	-0.3195J	-14.909	-31.616J	Stab Actuator
0.6985	0.2223J	-24.849	24.651J	TE Flap Actuator
0.6985	-0.2223J	-24.849	-24.651J	TE Flap Actuator
0.7597	-0.3344J	-14.903	-33.167J	Stab Actuator
0.7597	0.3344J	-14.903	33.167J	Stab Actuator
0.4907	-0.0613J	-56.336	-9.942J	Ail Actuator
0.4907	0.0613J	-56.336	9.942J	Ail Actuator
0.4120	0.0000J	-70.946	0.000J	Filter
0.4267	0.3259J	-49.749	52.187J	Ail Actuator
0.4267	-0.3259J	-49.749	-52.187J	Ail Actuator
0.4191	-0.3947J	-44.173	-60.434J	Rud Actuator
0.4191	0.3947J	-44.173	60.434J	Rud Actuator
0.4181	-0.3950J	-44.250	-60.555J	Rud Actuator
0.4181	0.3950J	-44.250	60.555J	Rud Actuator
0.3601	0.0000J	-81.707	0.000J	LE Flap Actuator
0.3601	0.0000J	-81.708	0.000J	LE Flap Actuator
0.2255	-0.4030J	-61.808	-84.844J	Stab Actuator
0.2255	0.4030J	-61.808	84.844J	Stab Actuator
0.2238	0.4019J	-62.114	85.016J	Stab Actuator
0.2238	-0.4019J	-62.114	-85.016J	Stab Actuator
0.1226	0.0000J	-167.885	0.000J	Rate Gyro Sensor
0.1216	0.0000J	-168.582	0.000J	Rate Gyro Sensor
0.1208	0.0000J	-169.101	0.000J	Rate Gyro Sensor
-0.0042	-0.0203J	-309.929	-141.910J	Accel Sensor
-0.0042	0.0203J	-309.929	141.910J	Accel Sensor
0.0017	-0.0082J	-382.321	-109.192J	Accel Sensor
0.0017	0.0082J	-382.321	109.192J	Accel Sensor
0.0033	0.0000J	-457.797	0.000J	Rate Gyro Sensor
0.0032	0.0000J	-460.647	0.000J	Rate Gyro Sensor
0.0031	0.0000J	-461.227	0.000J	Rate Gyro Sensor

TABLE 9
F/A-18A UNDAMAGED AIRCRAFT EIGENSTRUCTURE

Spiral		*Phugoid*		*Phugoid*	
λ = 0.0029 + 0.0000J		λ = -0.0173 + 0.0000j		λ = -0.5115 + 0.0000J	
0.0000	0.0000J	-1.0000	0.0000J	-0.3443	0.0000J
0.0000	0.0000J	0.0559	0.0000J	1.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0016	0.0000J
0.0000	0.0000J	0.0000	0.0000J	-0.0031	0.0000J
0.7764	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0494	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0029	0.0000J	0.0000	0.0000J	0.0000	0.0000J
1.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J

Roll		*Dutch Roll*		*Dutch Roll*	
λ = -2.9042		λ = -1.8703-2.6152J		λ = -1.8703 + 2.6152J	
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
-1.0000	0.0000J	-1.0000	-1.2610J	-1.0000	1.2610J
-0.0037	0.0000J	0.0026	-0.0075J	0.0026	0.0075J
-0.0478	0.0000J	0.0049	0.0166J	0.0049	-0.0166J
0.0165	0.0000J	-0.0051	-0.0018J	-0.0051	0.0018J

Short Period		*Short Period*	
λ = -2.2794-2.7016J		λ = -2.2794 + 2.7016J	
0.0183	-0.0031J	0.0183	0.0031J
-1.0000	-1.7738J	-1.0000	1.7738J
-0.0055	0.0078J	-0.0055	-0.0078J
-0.0007	-0.0026J	-0.0007	0.0026J
0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J

The initial test performed on the desired eigenstructure was the right null space constraint. Table 14 depicts the null space residuals for Case A. Note that RES_r is 0.036547 for this case which compares favorably with that of the undamaged (or nominal) value. Therefore the desired right eigenvectors were considered members of the Case A perturbed right null spaces and no reorientation was considered necessary. When the left eigenvectors were tested for left null space membership, a value of RES_l = 125.11 was obtained. When compared with a value of 3.3046, it was

TABLE 10

F/A-18A UNDAMAGED FEEDBACK GAIN MATRIX

	Q	*NZ*	*AA*	*R*
RST	0.214E+00	-0.131E+00	0.000E+00	0.000E+00
LST	0.214E+00	-0.131E+00	0.000E+00	0.000E+00
RLE	0.000E+00	0.000E+00	0.209E-01	0.000E+00
LLE	0.000E+00	0.000E+00	0.209E-01	0.000E+00
RTE	0.000E+00	0.000E+00	0.110E-01	0.000E+00
LTE	0.000E+00	0.000E+00	-0.110E-01	0.000E+00
RA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RR	0.000E+00	0.000E+00	0.000E+00	0.689E+00
LR	0.000E+00	0.000E+00	0.000E+00	0.689E+00

	P	*NY*	*C1*	*C2*
RST	0.240E-01	0.000E+00	-0.257E-02	0.748E-03
LST	0.240E-01	0.000E+00	-0.257E-02	0.748E-03
RLE	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LLE	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RTE	0.192E-01	0.000E+00	0.000E+00	0.000E+00
LTE	-0.192E-01	0.000E+00	0.000E+00	0.000E+00
RA	0.600E-01	0.000E+00	0.000E+00	0.000E+00
LA	-0.600E-01	0.000E+00	0.000E+00	0.000E+00
RR	0.388E+00	0.165E+02	0.000E+00	0.000E+00
LR	0.388E+00	0.165E+02	0.000E+00	0.000E+00

	C3	*C4*	*C5*	*C6*
RST	-0.191E-01	-0.211E+00	-0.383E-01	0.000E+00
LST	-0.191E-01	-0.211E+00	-0.383E-01	0.000E+00
RLE	0.000E+00	0.000E+00	0.000E+00	0.412E-01
LLE	0.000E+00	0.000E+00	0.000E+00	0.412E-01
RTE	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LTE	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RR	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LR	0.000E+00	0.000E+00	0.000E+00	0.000E+00

	C7	*C8*	*C9*	*C10*
RST	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LST	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RLE	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LLE	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RTE	0.219E-01	0.000E+00	0.000E+00	0.000E+00
LTE	0.219E-01	0.000E+00	0.000E+00	0.000E+00
RA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RR	0.000E+00	-0.856E-02	-0.494E-02	0.000E+00
LR	0.000E+00	-0.856E-02	-0.494E-02	0.000E+00

	C11	*C12*
RST	0.000E+00	0.000E+00
LST	0.000E+00	0.000E+00
RLE	0.000E+00	0.000E+00
LLE	0.000E+00	0.000E+00
RTE	0.000E+00	0.000E+00
LTE	0.000E+00	0.000E+00
RA	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00
RR	0.317E-02	0.000E+00
RR	0.317E-02	0.000E+00

TABLE 11

F/A-18A NOMINAL NULL SPACE RESIDUALS

$$RES_r = \sum \|\varepsilon_{ri}\|_2 = 0.74838E-01$$

$$RES_l = \sum \|\varepsilon_{li}\|_2 = 0.33046E+01$$

Mode	$\|\varepsilon_{ri}\|$		$\|\varepsilon_{li}\|$	
	2-Norm	∞-Norm	2-Norm	∞norm
ShPer	0.10646E-04	0.71796E-05	0.52883E+00	0.11033E-03
ShPer	0.10646E-04	0.71796E-05	0.52883E+00	0.11033E-03
P11D	0.10902E-04	0.76087E-05	0.14053E-01	0.44177E-05
DRoll	0.30222E-02	0.29889E-02	0.41392E-03	0.70056E-13
DRoll	0.30222E-02	0.29889E-02	0.41392E-03	0.70056E-13
FPhug	0.36685E-05	0.23228E-05	0.16863E+00	0.33777E-04
Y3D	0.43926E-05	0.29008E-05	0.76362E-01	0.13423E-04
Roll	0.53555E-03	0.52682E-03	0.31910E-03	0.17306E-12
SPhug	0.38242E-06	0.26453E-06	0.12615E+01	0.12600E+01
Filtr	0.53746E-04	0.40515E-04	0.10055E-03	0.11079E-14
Filtr	0.33415E-03	0.32759E-03	0.49189E-03	0.14597E-14
Filtr	0.16079E-04	0.13604E-04	0.69333E+00	0.12940E-05
Filtr	0.28549E-11	0.20861E-11	0.13364E-03	0.33852E-15
Filtr	0.14593E-12	0.96229E-13	0.23156E-01	0.56116E-08
Filtr	0.45935E-03	0.45639E-03	0.80447E-02	0.65734E-15
Spira	0.24578E-10	0.17275E-10	0.00000E+00	0.00000E+00
Filtr	0.48253E-12	0.38527E-12	0.00000E+00	0.00000E+00
Filtr	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00

decided that reorienting the left eigenvectors was required. In particular, note that the source of left residual error was predominantly due to the short period, phugoid, and two of the filter modes, P11D and Y3D. P11D is a longitudinal filter which sends angle of attack data to the collective flap commands. The Y3D is a lateral feedback filter which smooths yaw and roll rate feedback to the synchronous rudder commands. Note that the lateral aircraft modes satisfy both right and left perturbed null space constraints as might be expected since the degradation is in the longitudinal surfaces only. Note additionally that the three modes at the bottom of Table 14 have true zero left residuals just as in the undamaged case. This depicts the dependence of the residual on the open loop parameters (A,B,C) and show that for some eigenvalues, the null space constraints will always be satisfied regardless of the orientation of the eigenvector. The converse is also true, some eigenvalues will always invoke a null space residual. The key is to orient their vectors as close as possible to the desired orientation while minimizing the residual error.

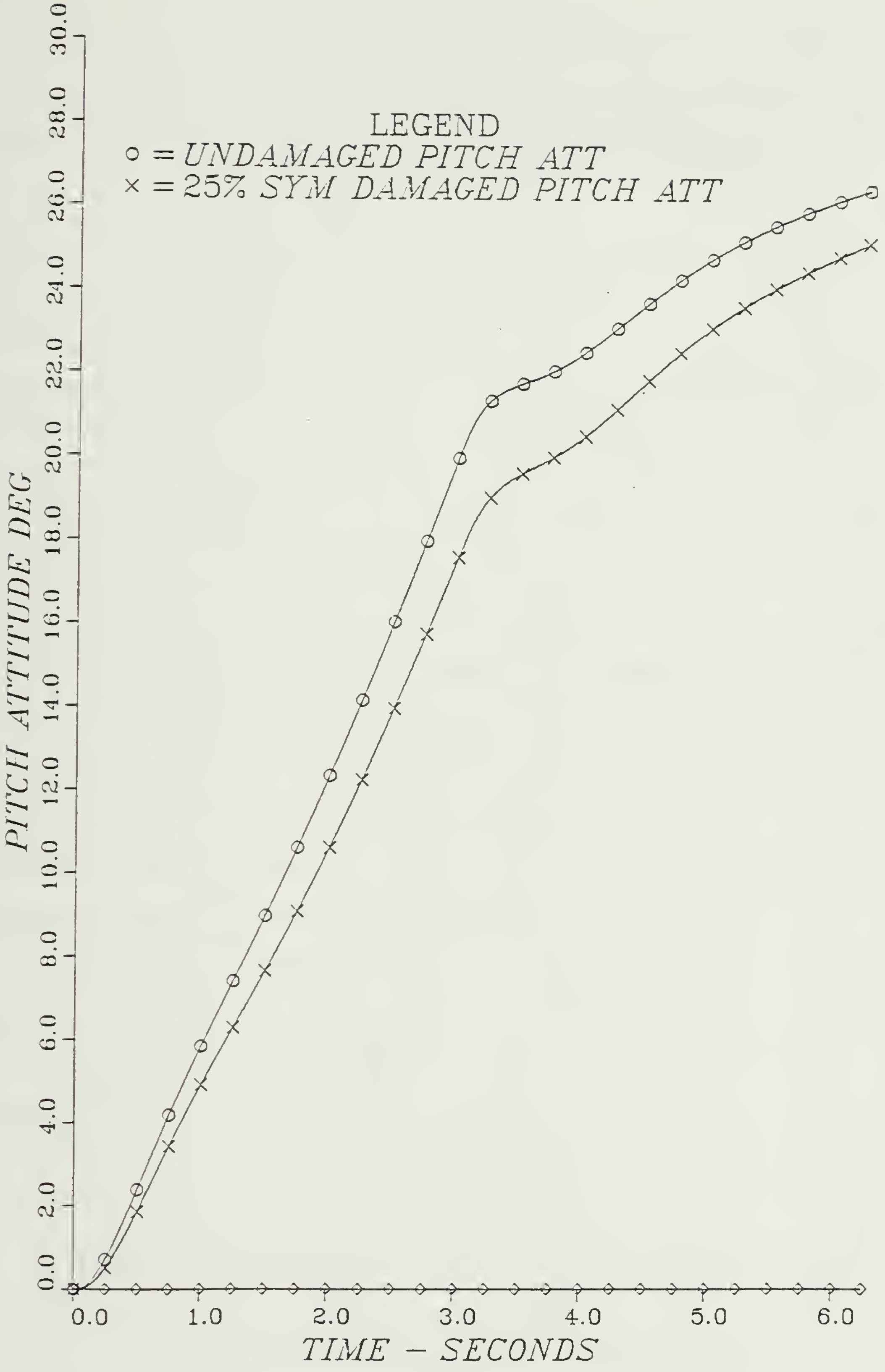


Figure 8.6 Case A Damaged Pitch Attitude Response.

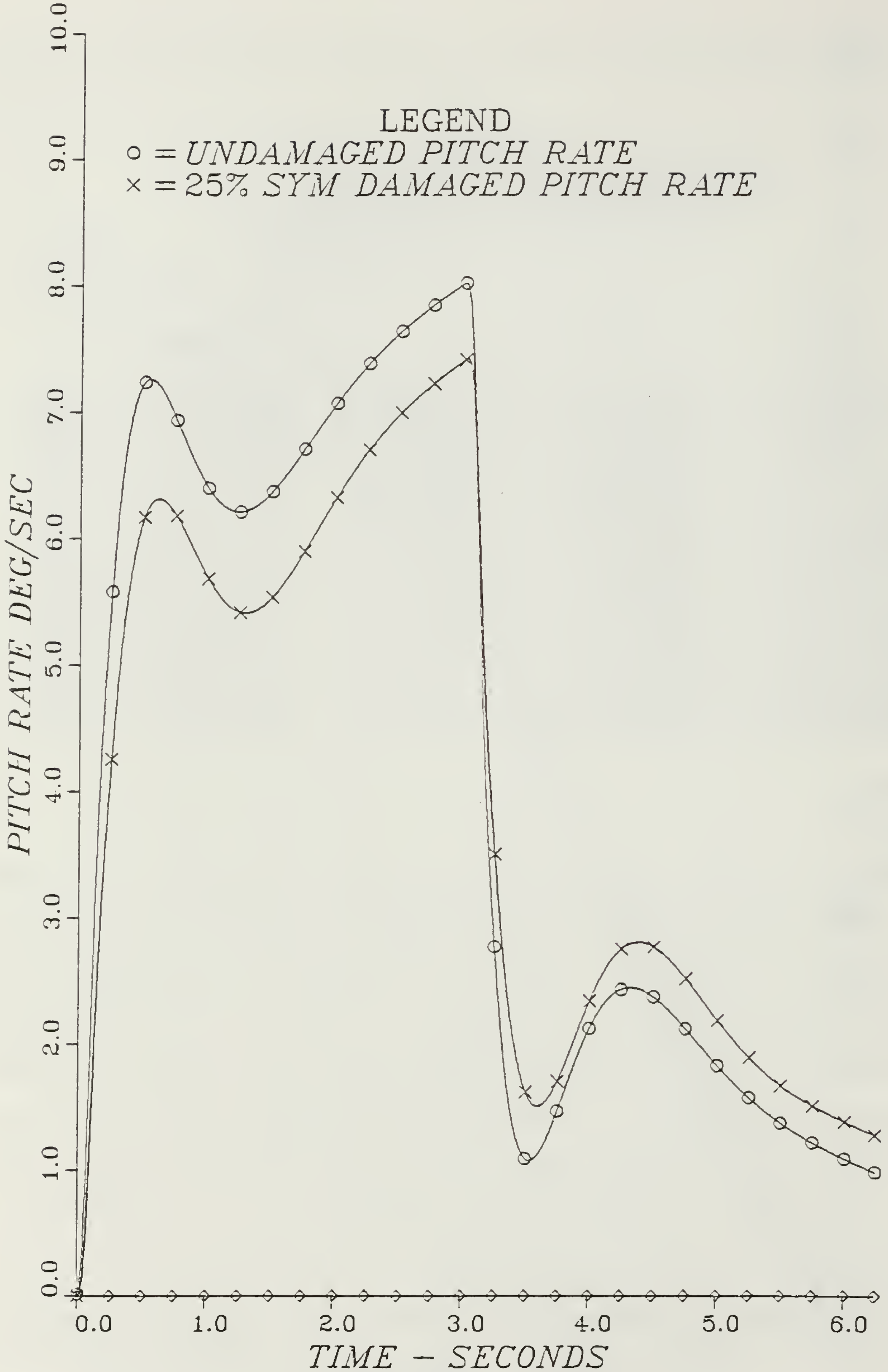


Figure 8.7 Case A Damaged Pitch Rate Response.

Minimization of the euclidean norm of the left null vector, ε_{li}, for each of these four modes was performed in the following way. All fifty-five elements of the left eigenvector were allowed to vary ±0.001 of their initial value to provide freedom to rotate as close as possible to the null space invoked by the fixed eigenvalue. ADS was then used to minimize $\|\varepsilon_l\|_2$ individually for each of the four modes. Upon completion of the minimization procedure, final values of $RES_r = 0.043691$ and $RES_l = 37.117$ were obtained. Note that the right residual sum maintained an acceptable value during minimization of the left null vector norms. This is demonstrative of the higher dimension of the right null space.($d(E_r) = p = 45$) as compared with that of the left null space ($d(E_l) = q = 37$). Subsequent to the minimization, the fast eigenspace was preserved by inserting the undamaged elements 9-43,45, and 47 into the 37 fast modes of the right eigenvector matrix. The reconfigured feedback gains were then computed and responses were plotted. In essence, this is a reduced order design method and proved successful for the symmetric degradation cases.

Table 15 shows the reconfigured feedback gains for Case A. Note that the reconfigured gains invoke deflection commands to the leading and trailing edge flaps via pitch rate, normal acceleration, and angle of attack feedback paths. Additionally, feedback gains are now present in the angle of attack to stabilator feedback path. Note that these paths are not present in the longitudinal control model of Figure 8.3. This solution therefore would require a modification to the aircraft control system. Additionally there are feedback gains between selected filter states and the leading/trailing edge surfaces. These particular filter states correspond to the pitch rate to collective stabilator, normal acceleration to collective stabilator, pitch stick to collective stabilator, and angle of attack to collective leading flap feedback filters. This longitudinal reconfiguration was implicitly invoked by the design technique described above. Table 16 depicts the reconfigured eigenstructure for Case A. Note the slower phugoid ($\lambda = -0.4616$) has been shifted to 90.24% of the undamaged value which is a 9.59% improvement over the damaged eigenvalue location. Also noteworthy is the weak modal coupling to forward and vertical velocity in the spiral mode. Similarly, there is weak coupling to side velocity in the reconfigured short period. Figures 8.8 amd 8.9 show the reconfigured pitch attitude and pitch rate responses. Note that the undamaged responses were not exactly regained, but the reconfigured responses show positive signs toward performance recovery. This is indicative of satisfying the null space constraints while not invoking a constraint on the orientation of the right

TABLE 12

CASE A (25% SYMMETRIC) DAMAGED C-LOOP EIGENVALUES

Z-Plane Roots		*S-Plane Roots*		*Mode*
1.0000	0.0000J	0.000	0.000J	
1.0000	0.0000J	0.000	0.000J	
1.0000	*0.0000J*	*0.003*	*0.000J*	*Spiral*
1.0000	0.0000J	0.003	0.000J	
0.9998	*0.0000J*	*-0.019*	*0.000J*	*Phugoid*
0.9949	*0.0000J*	*-0.412*	*0.000J*	*Phugoid*
0.9876	0.0000J	-1.000	0.000J	
0.9863	0.0000J	-1.104	0.000J	
0.9821	0.0000J	-1.443	0.000J	
0.9753	0.0000J	-2.000	0.000J	
0.9753	0.0000J	-2.000	0.000J	
0.9753	0.0000J	-2.004	0.000J	
0.9645	*0.0000J*	*-2.892*	*0.000J*	*Roll*
0.9764	*-0.0319J*	*-1.869*	*-2.615J*	*Dutch Roll*
0.9764	*0.0319J*	*-1.869*	*2.615J*	*Dutch Roll*
0.9575	0.0000J	-3.477	0.000J	
0.9759	*-0.0361J*	*-1.899*	*-2.956J*	*Short Period*
0.9759	*0.0361J*	*-1.899*	*2.956J*	*Short Period*
0.8406	0.0000J	-13.886	0.000J	
0.7519	0.0000J	-22.809	0.000J	
0.7109	0.0000J	-27.292	0.000J	
0.7106	0.0000J	-27.328	0.000J	
0.6988	-0.2220J	-24.821	-24.612J	
0.6988	0.2220J	-24.821	24.612J	
0.6985	-0.2223J	-24.849	-24.651J	
0.6985	0.2223J	-24.849	24.651J	
0.7642	0.3236J	-14.915	32.043J	
0.7642	-0.3236J	-14.915	-32.043J	
0.7596	-0.3344J	-14.908	-33.175J	
0.7596	0.3344J	-14.908	33:175J	
0.4907	-0.0613J	-56.335	-9.938J	
0.4907	0.0613J	-56.335	9.938J	
0.4119	0.0000J	-70.956	0.000J	
0.4267	0.3259J	-49.749	52.187J	
0.4267	-0.3259J	-49.749	-52.187J	
0.4191	0.3947J	-44.173	60.434J	
0.4191	-0.3947J	-44.173	-60.434J	
0.4181	0.3950J	-44.250	60.555J	
0.4181	-0.3950J	-44.250	-60.555J	
0.3601	0.0000J	-81.707	0.000J	
0.3601	0.0000J	-81.708	0.000J	
0.2251	-0.4027J	-61.888	-84.888J	
0.2251	0.4027J	-61.888	84.888J	
0.2238	0.4019J	-62.117	85.018J	
0.2238	-0.4019J	-62.117	-85.018J	
0.1226	0.0000J	-167.882	0.000J	
0.1219	0.0000J	-168.385	0.000J	
0.1208	0.0000J	-169.101	0.000J	
-0.0042	-0.0203J	-309.929	-141.910J	
-0.0042	0.0203J	-309.929	141.910J	
0.0017	-0.0083J	-381.587	-109.565J	
0.0017	0.0083J	-381.587	109.565J	
0.0032	0.0000J	-458.818	0.000J	
0.0032	0.0000J	-460.648	0.000J	
0.0031	0.0000J	-461.242	0.000J	

TABLE 13

CASE A (25%SYMMETRIC DAMAGE) AIRCRAFT EIGENSTRUCTURE

Spiral		*Phugoid*		*Phugoid*	
λ =0.0029		λ = -0.0187 + 0.0000J		λ = -0.4123-0.0000J	
0.0000	0.0000J	1.0000	0.0000J	0.4938	0.0000J
0.0000	0.0000J	-0.0564	0.0000J	-1.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	-0.0016	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0039	0.0000J
-0.7765	0.0000J	0.0000	0.0000J	0.0000	0.0000J
-0.0494	0.0000J	0.0000	0.0000J	0.0000	0.0000J
-0.0029	0.0000J	0.0000	0.0000J	0.0000	0.0000J
-1.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J

Roll		*Dutch Roll*		*Dutch Roll*	
λ = -2.8919		λ = -1.8691-2.6153J		λ = -1.8691 + 2.6153J	
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
1.0000	0.0000J	1.0000	-60.474J	1.0000	60.474J
0.0037	0.0000J	0.2549	-0.1567J	0.2549	0.1567J
0.0475	0.0000J	-0.2563	0.5983J	-0.2563	-0.5983J
-0.0164	0.0000J	-0.1051	-0.1731J	-0.1051	0.1731J

Short Period		*Short Period*	
λ = -1.8994-2.9561J		λ = -1.8994 + 2.9561J	
0.0056	-0.0083J	0.0056	0.0083J
-1.0000	-0.4514J	-1.0000	0.4514J
-0.0007	0.0052J	-0.0007	-0.0052J
-0.0011	-0.0010J	-0.0011	0.0010J
0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J

eigenvector. Although the reconfigured gains provide positive steps toward regaining the undamaged response, full performance recovery was not obtained due to the undesirable orientation of the longitudinal right eigenvectors.

2. Case B

Figures 8.10 and 8.11 show the damaged longitudinal responses for the 50% symmetric degradation case. Tables 17 and 18 depict the perturbed full order and aircraft eigenstructures respectively. Table 19 shows the null space residuals. Note

TABLE 14

CASE A NULL SPACE RESIDUALS

$$RES_r = \sum \|\varepsilon_{ri}\| = 0.36547E-01$$

$$RES_l = \sum \|\varepsilon_{li}\| = 0.12511E+03$$

Mode	$\|\varepsilon_{ri}\|_2$		$\|\varepsilon_{li}\|_2$	
	2-Norm	∞-Norm	2-Norm	∞norm
ShPer	0.34926E-02	0.31823E-02	0.33484E+02	0.23673E+02
ShPer	0.34926E-02	0.31823E-02	0.33484E+02	0.23673E+02
P11D	0.29547E-02	0.26955E-02	0.37790E+02	0.26721E+02
DRoll	0.43344E-02	0.36901E-02	0.90522E-02	0.70056E-13
DRoll	0.43344E-02	0.36901E-02	0.90522E-02	0.70056E-13
FPhug	0.19289E-02	0.17545E-02	0.10999E+02	0.77762E+01
Y3D	0.21797E-02	0.19839E-02	0.68938E+01	0.48743E+01
Roll	0.43476E-02	0.37847E-02	0.10488E+00	0.17306E-12
SPhug	0.29235E-03	0.26580E-03	0.15057E+01	0.12600E+01
Filtr	0.68759E-04	0.52844E-04	0.20671E-01	0.11079E-14
Filtr	0.14587E-02	0.13553E-02	0.11536E-01	0.14597E-14
Filtr	0.72043E-02	0.65518E-02	0.69335E+00	0.12940E-05
Filtr	0.28572E-11	0.20878E-11	0.27317E-01	0.33852E-15
Filtr	0.25527E-12	0.17498E-12	0.23157E-01	0.56116E-08
Filtr	0.45755E-03	0.45448E-03	0.51955E-01	0.65734E-15
Spira	0.89445E-10	0.78303E-10	0.00000E+00	0.00000E+00
Filtr	0.49205E-12	0.38527E-12	0.00000E+00	0.00000E+00
Filtr	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00

that the *bad actors* are again the short period, fast phugoid, and the same filter modes as Case A. Additionally note the doubling of the residual sums from Case A to Case B. The design procedure noted above for Case A was repeated for this case. Tables 20 and 21 show the reconfigured feedback gains and aircraft eigenstructure for Case B. Figures 8.12 and 8.13 show the reconfigured responses.

The lack of full performance recovery is more clearly demonstrated in these reconfigured responses. Note that although the reconfigured phugoid (λ = -0.3699) eigenvalue is a 26.25% improvement over the damaged eigenvalue position (λ = -0.293), it is 27.73% slower than the undamaged value of -0.511. The reconfigured short period eigenvalue (λ = -1.505 ±j3.013) has not been shifted and has less modal coupling to vertical velocity (-1.0000 ±j0.8656) than the undamaged short period modes (-1.0000 ±j1.17738). The response curves also indicate that although the null space constraints have been met, performance has not been fully recovered due, in part, to the reorientation of the phugoid and short period right eigenvectors.

TABLE 15

RECONFIGURED FEEDBACK GAINS FOR CASE A DAMAGE

	Q	NZ	AA	R
RST	0.214E+00	-0.131E+00	0.459E-03	0.000E+00
LST	0.214E+00	-0.131E+00	0.459E-03	0.000E+00
RLE	-0.355E-02	0.745E-02	0.267E-01	0.000E+00
LLE	-0.355E-02	0.745E-02	0.267E-01	0.000E+00
RTE	0.608E-02	-0.959E-02	0.462E-02	0.000E+00
LTE	0.608E-02	-0.959E-02	0.462E-02	0.000E+00
RA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RR	0.000E+00	0.000E+00	0.000E+00	0.689E+00
LR	0.000E+00	0.000E+00	0.000E+00	0.689E+00
	P	NY	C1	C2
RST	0.240E-01	0.000E+00	-0.261E-02	0.747E-03
LST	-0.240E-01	0.000E+00	-0.261E-02	0.747E-03
RLE	0.000E+00	0.000E+00	-0.502E-03	0.000E+00
LLE	0.000E+00	0.000E+00	-0.502E-03	0.000E+00
RTE	0.192E-01	-0.264E-03	0.531E-03	0.000E+00
LTE	-0.192E-01	0.000E+00	0.531E-03	0.000E+00
RA	0.600E-01	0.000E+00	0.000E+00	0.000E+00
LA	-0.600E-01	0.000E+00	0.000E+00	0.000E+00
RR	0.388E+00	0.165E+02	0.000E+00	0.000E+00
LR	0.388E+00	0.165E+02	0.000E+00	0.000E+00
	C3	C4	C5	C6
RST	-0.194E-01	-0.210E+00	-0.387E-01	0.000E+00
LST	-0.194E-01	-0.210E+00	-0.387E-01	0.000E+00
RLE	-0.373E-02	0.941E-02	-0.746E-02	0.413E-01
LLE	-0.373E-02	0.941E-02	-0.746E-02	0.413E-01
RTE	0.395E-02	-0.125E-01	0.790E-02	0.000E+00
LTE	0.395E-02	-0.125E-01	0.790E-02	0.000E+00
RA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RR	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LR	0.000E+00	0.000E+00	0.000E+00	0.000E+00
	C7	C8	C9	C10
RST	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LST	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RLE	0.436E-03	0.000E+00	0.000E+00	0.000E+00
LLE	0.436E-03	0.000E+00	0.000E+00	0.000E+00
RTE	0.214E-01	0.000E+00	0.000E+00	0.000E+00
LTE	0.214E-01	0.000E+00	0.000E+00	0.000E+00
RA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RR	0.000E+00	-0.856E-02	-0.494E-02	0.000E+00
LR	0.000E+00	-0.856E-02	-0.494E-02	0.000E+00
	C11	C12		
RST	0.000E+00	0.000E+00		
LST	0.000E+00	0.000E+00		
RLE	0.000E+00	0.000E+00		
LLE	0.000E+00	0.000E+00		
RTE	0.000E+00	0.000E+00		
LTE	0.000E+00	0.000E+00		
RA	0.000E+00	0.000E+00		
LA	0.000E+00	0.000E+00		
RR	0.318E-02	0.000E+00		
LR	0.318E-02	0.000E+00		

TABLE 16
CASE A RECONFIGURED AIRCRAFT EIGENSTRUCTURE

Spiral		*Phugoid*		*Phugoid*	
$\lambda = 0.0029$		$\lambda = -0.0183+0.0000J$		$\lambda = -0.4616-0.0000J$	
0.0047	0.0000J	1.0000	0.0000J	-0.4260	0.0000J
-0.0004	0.0000J	-0.0549	0.0000J	1.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0017	0.0000J
0.0000	0.0000J	0.0000	0.0000J	-0.0037	0.0000J
-0.7769	0.0000J	0.0000	0.0000J	-0.0002	0.0000J
-0.0494	0.0000J	0.0000	0.0000J	0.0000	0.0000J
-0.0029	0.0000J	0.0000	0.0000J	0.0000	0.0000J
-1.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J

Roll		*Dutch Roll*		*Dutch Roll*	
$\lambda = -2.8919$		$\lambda = -1.8693-2.6152J$		$\lambda = -1.8693+2.6152J$	
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
-1.0000	0.0000J	-1.0000	-0.0130J	-1.0000	0.0130J
-0.0037	0.0000J	-0.0026	-0.0042J	-0.0026	0.0042J
-0.0475	0.0000J	0.0099	0.0042J	0.0099	-0.0042J
0.0164	0.0000J	-0.0029	0.0017J	-0.0029	-0.0017J

Short Period		*Short Period*	
$\lambda = -1.8758-2.9291J$		$\lambda = -1.8758+2.9291J$	
-0.0222	-0.0128J	-0.0222	0.0128J
-1.0000	-2.5686J	-1.0000	2.5686J
0.0129	0.0014J	0.0129	-0.0014J
-0.0023	-0.0029J	-0.0023	0.0029J
-0.0014	-0.0010J	-0.0014	0.0010J
0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J

The design procedure for Case B was therefore modified to enhance performance recovery in the following way. Inspection of the left residual null vector. ε_{li}, for the short period, phugoid, and P11D filter modes revealed that the residuals were predominantly due to the real part of the null vector. In order to reduce the norm of the residuals, the real parts of fast elements (9-43,45,47) were therefore unrestricted during the minimization. In this way the elements of the slow left eigenvector which corresponded to the faster actuator and sensor modes, were allowed to assume the dominant role in the orientation. Upon completion of this minimization,

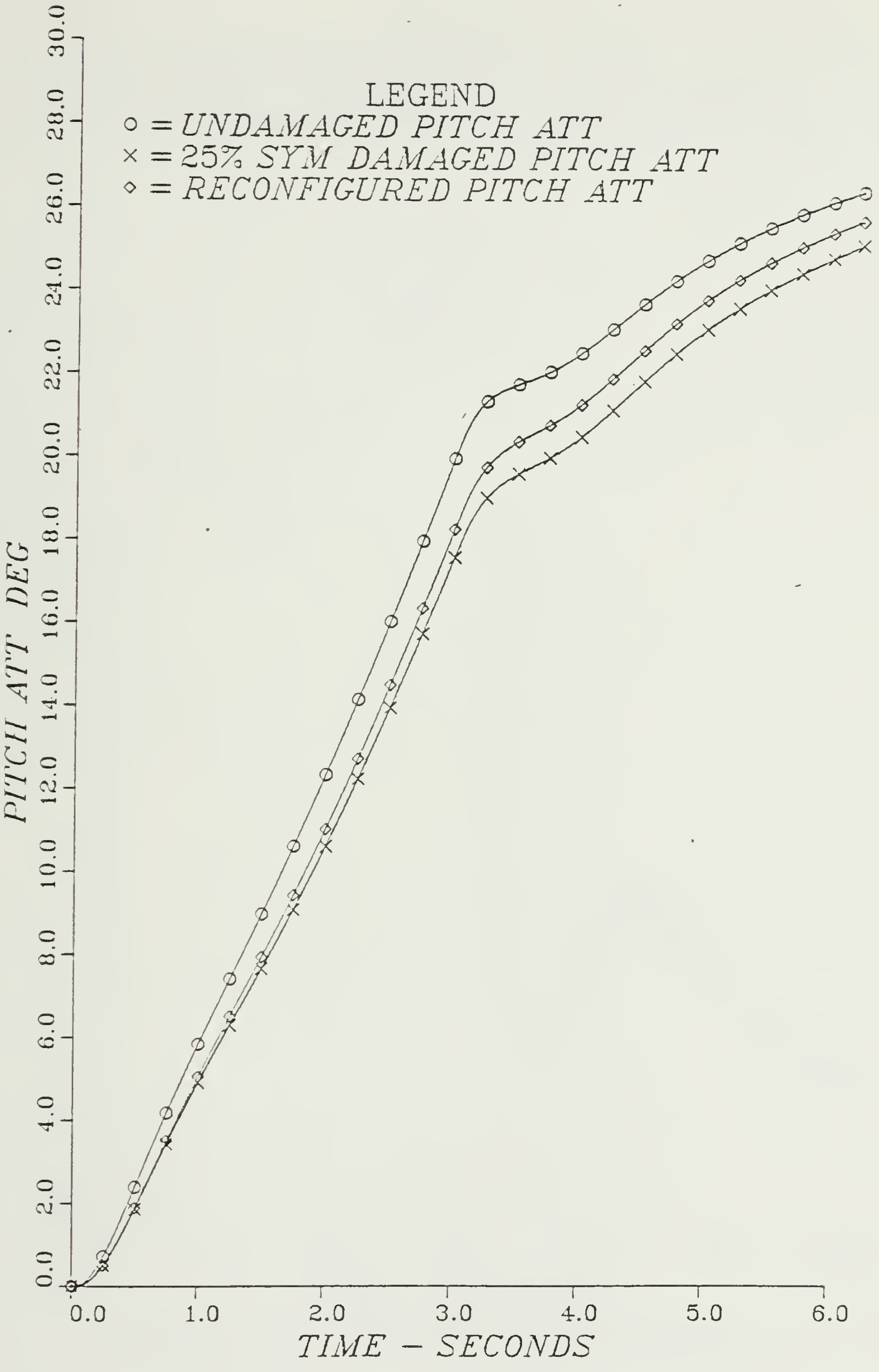


Figure 8.8. Case A Reconfigured Pitch Attitude Response.

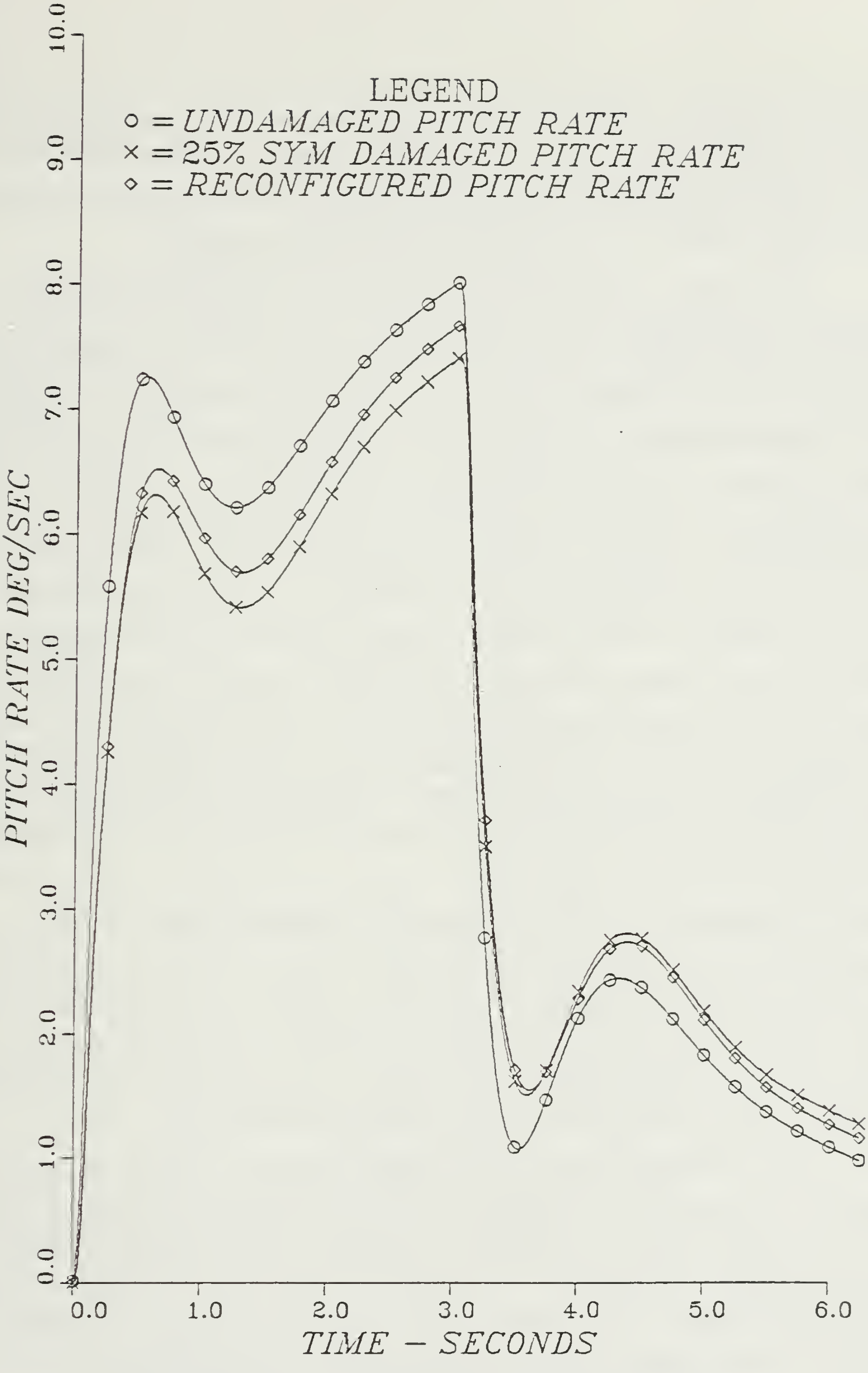


Figure 8.9 Case A Reconfigured Pitch Rate Response.

final values of RES_r = 0.07127 and RES_l = 72.7 were obtained. Again note the right residual sum, RES_r, indicates that the new right eigenvectors remain in the allowable right null space. The fast eigenspace was inserted into the thirty-seven fast modes in a similar fashion as before and the reconfigured gains were recomputed.

The resulting eigenstructure revealed that performance had improved somewhat but additional recovery was required. The fast phugoid, short period, and P9D filter modes exhibited short falls when compared to the undamaged modal structure. As noted previously, a procedure at this point would be to compare the present right eigenvectors with those which give rise to performance recovery but perhaps a penalty in the eigenvalue location. One such design was performed in the following fashion. By modifying the objective function during the ADS minimization to represent the absolute value of the sum of the components of the left null vector, ε_{li}, a different set of right eigenvectors was computed. As these particular eigenvectors gave rise to a RES_l = 199.5 and RES_r = 0.0066, the resulting closed loop eigenvalues were not exactly invoked due to the value of RES_l . However, the response of the system exhibited better performance recovery. Upon comparing the slow phugoid, short period, and P9D filter modal structures with these better performing structures, noted differences in the structure were evident. Therefore, the right eigenvector modes for the slow phugoid, short period, and P9D filter were replaced with the better performing vectors which gave rise to better performance recovery. The new reconfigured gains and aircraft eigenstructure are shown in Tables 22 and 23. The responses are shown in Figures 8.14 and 8.15. Note the shift of the phugoid to -0.43006 and the modal structure of the short period. The slow phugoid has been shifted to 84.1% of the undamaged value vice 72.4% in the first Case B design. This is a 26.8% increase over the damaged slow phugoid location vice a 15.1% increase in the first design. Note the improved responses in Figures 8.14 and 8.15.

This particular method of performance recovery is not to be taken as a standard by any means. It is simply one which was successful for the design problem at hand. Further research would entail developing a more systematic procedure for recovering desired performance. For example, if one were to simply insert the undamaged modal structures for the slow phugoid, short period, and P9D filter, performance would improve but at the expense of the slow eigenvalue locations.

In summary, one converged on an allowable eigenvector set which will invoke eigenvalue locations via the feedback gain equation but lacks full performance recovery in the short period and slow phugoid modes. Performance recovery was obtained by comparing modal short falls with a right eigenvector set which was known to give rise to better performance recovery and then inserting the better performing modal structures. Feedback gains were then recomputed and the responses improved.

F. THE ASYMMETRIC DEGRADATION PROBLEM

Asymmetric stabilator degradation was simulated by linearly decreasing only the right stabilator control effectiveness via appropriate modification of the gain matrix. Tables 24 and 25 show the perturbed full order eigenvalues and aircraft eigenstructure respectively. The predominant change in the location of the eigenvalues occur in the slower phugoid (λ = - 0.464) and short period modes which is the same pattern as Case A. The significant difference however between Case C and the symmetric cases lie in the eigenvectors of the aircraft modes. Note the modal cross coupling for all the aircraft modes in Table 25. The significant cross coupling terms are noted as the roll mode coupling to vertical velocity, the short period coupling to side velocity, the spiral mode coupling to forward velocity, and dutch roll mode coupling to vertical velocity. Figures 8.14 through 8.17 show the resulting damaged responses to the 1.0 in 3.0 sec longitudinal command. The longitudinal responses are rather benign, however the resulting undesired bank angle and yaw rate responses are clearly depicted. This bank angle and yaw rate coupling was of course not present for the symmetric degradation cases.

A solution method posed for this problem involves a superposition technique. Since the 25% symmetric case has been solved, one already has computed the gains which reconfigure such a system. A superposition technique would allow one to use the portion of these gains which would symmetrically balance the asymmetric control deficiency. Once the aircraft has regained its undamaged response, these balancing gains would then be removed and the gain values would be reset to their unperturbed values. The decision as to whether the aircraft had regained its undamaged longitudinal structure could then be based on the presence of roll or yaw. If one used, for example, bank angle sensing as decision criteria, one in effect has designed a roll detection reconfiguration controller. This type of controller was used as a solution technique. Note that this technique is actually a digital switch which is evident upon examining the reconfigured responses in Figures 8.16 through 8.19. A system advantage of such a controller might be that one would be required to store symmetric

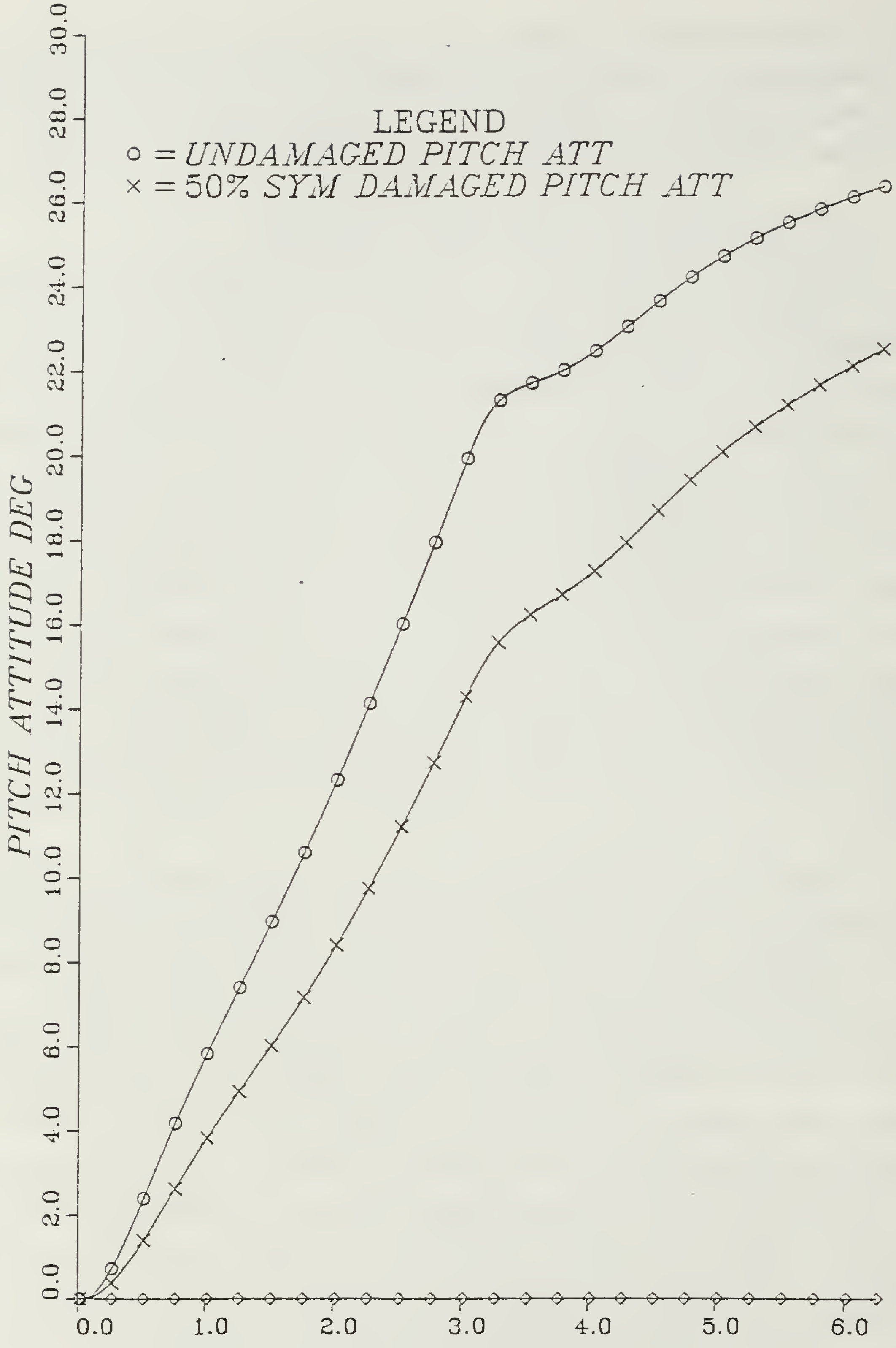


Figure 8.10 Case B Damaged Pitch Attitude Response.

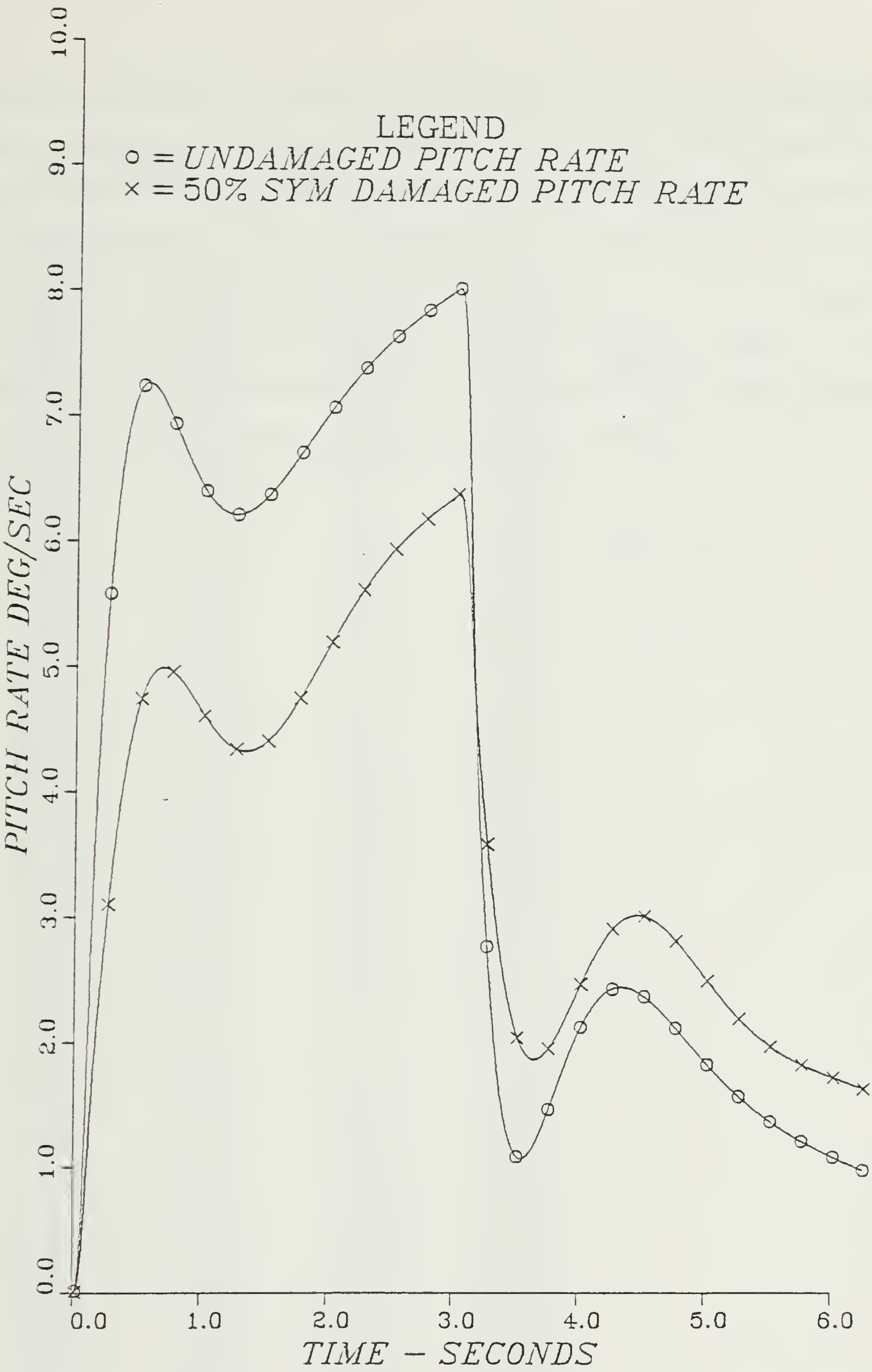


Figure 8.11 Case B Damaged Pitch Rate Response.

reconfiguration gains only. Asymmetric reconfiguration could then be invoked by a roll detection switch. Figures 8.16 through 8.17 depict the reconfigured responses subsequent to such a reconfiguration technique. Table 24 shows the reconfigured gains which are invoked during the time the aircraft is experiencing unacceptable bank angles. Note that the bank angle response, although oscillatory, exhibits far less coupling. The yaw rate response is also oscillatory and has decreased in amplitude. Additional research would involve investigating the origin of the oscillation and perform a smoothing of the responses. The important item here is that the technique exhibits successful regaining of the undamaged bank angle response. One would then be able to solve several asymmetric problems using the corresponding symmetric solutions.

TABLE 17

CASE B(50% SYMMETRIC) DAMAGED C-LOOP EIGENVALUES

Z-Plane Roots		*S-Plane Roots*		
1.0000	0.0000J	0.000	0.000J	
1.0000	0.0000J	0.000	0.000J	
1.0000	*0.0000J*	*0.003*	*0.000J*	*Spiral*
1.0000	0.0000J	0.003	0.000J	
0.9997	*0.0000J*	*-0.022*	*0.000J*	*Phugoid*
0.9963	*0.0000J*	*-0.293*	*0.000J*	*Phugoid*
0.9876	0.0000J	-1.000	0.000J	
0.9868	0.0000J	-1.066	0.000J	
0.9821	0.0000J	-1.442	0.000J	
0.9753	0.0000J	-2.000	0.000J	
0.9753	0.0000J	-2.000	0.000J	
0.9753	0.0000J	-2.004	0.000J	
0.9646	*0.0000J*	*-2.880*	*0.000J*	*Roll*
0.9764	*-0.0319J*	*-1.868*	*-2.615J*	*Dutch Roll*
0.9764	*0.0319J*	*-1.868*	*2.615J*	*Dutch Roll*
0.9595	0.0000J	-3.306	0.000J	
0.9804	*-0.0375J*	*-1.527*	*-3.056J*	*Short Period*
0.9804	*0.0375J*	*-1.527*	*3.056J*	*Short Period*
0.8403	0.0000J	-13.922	0.000J	
0.7435	0.0000J	-23.710	0.000J	
0.7109	0.0000J	-27.292	0.000J	
0.7107	0.0000J	-27.326	0.000J	
0.6988	-0.2220J	-24.821	-24.612J	
0.6988	0.2220J	-24.821	24.612J	
0.6985	-0.2223J	-24.848	-24.652J	
0.6985	0.2223J	-24.848	24.652J	
0.7625	-0.3274J	-14.920	-32.447J	
0.7625	0.3274J	-14.920	32.447J	
0.7595	0.3345J	-14.914	33.183J	
0.7595	-0.3345J	-14.914	-33.183J	
0.4907	-0.0612J	-56.334	-9.934J	
0.4907	0.0612J	-56.334	9.934J	
0.4119	0.0000J	-70.965	0.000J	
0.4267	0.3259J	-49.749	52.187J	
0.4267	-0.3259J	-49.749	-52.187J	
0.4191	0.3947J	-44.173	60.434J	
0.4191	-0.3947J	-44.173	-60.434J	
0.4181	-0.3950J	-44.250	-60.555J	
0.4181	0.3950J	-44.250	60.555J	
0.3601	0.0000J	-81.707	0.000J	
0.3601	0.0000J	-81.708	0.000J	
0.2247	0.4024J	-61.967	84.933J	
0.2247	-0.4024J	-61.967	-84.933J	
0.2238	0.4019J	-62.120	85.019J	
0.2238	-0.4019J	-62.120	-85.019J	
0.1226	0.0000J	-167.878	0.000J	
0.1222	0.0000J	-168.188	0.000J	
0.1208	0.0000J	-169.101	0.000J	
-0.0042	-0.0203J	-309.929	-141.910J	
-0.0042	0.0203J	-309.929	141.910J	
0.0017	0.0084J	-380.871	109.936J	
0.0017	-0.0084J	-380.871	-109.936J	
0.0032	0.0000J	-459.806	0.000J	
0.0032	0.0000J	-460.650	0.000J	
0.0031	0.0000J	-461.257	0.000J	

TABLE 18

CASE B (50% SYMMETRIC DAMAGE) AIRCRAFT EIGENSTRUCTURE

Spiral		*Phugoid*		*Phugoid*	
$\lambda = 0.0029 + 0.0000J$		$\lambda = -0.0216 + 0.0000J$		$\lambda = -0.2929 + 0.000J$	
0.0000	0.0000J	-1.0000	0.0000J	0.8985	0.0000J
0.0000	0.0000J	0.0575	0.0000J	-1.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	-0.0016	0.0000J
0.0000	0.0000J	-0.0001	0.0000J	0.0056	0.0000J
0.7766	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0494	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0029	0.0000J	0.0000	0.0000J	0.0000	0.0000J
1.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J

Roll		*Dutch Roll*		*Dutch Roll*	
$\lambda = -2.8796 + 0.0000J$		$\lambda = -1.8680 - 2.6153J$		$\lambda = -1.8680 + 2.6153J$	
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
1.0000	0.0000J	1.0000	-0.8315J	1.0000	0.8315J
0.0037	0.0000J	0.0061	0.0020J	0.0061	-0.0020J
0.0473	0.0000J	-0.0134	0.0041J	-0.0134	-0.0041J
-0.0164	0.0000J	0.0014	-0.0041J	0.0014	0.0041J

Short Period		*Short Period*	
$\lambda = -1.5271 - 3.0560J$		$\lambda = -1.5271 + 3.0560J$	
0.0300	0.0098J	0.0300	-0.0098J
1.0000	-3.2090J	1.0000	3.2090J
-0.0157	-0.0025J	-0.0157	0.0025J
0.0027	-0.0038J	0.0027	0.0038J
0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J

TABLE 19

CASE B NULL SPACE RESIDUALS

$$RES_r=\sum\|\varepsilon_{ri}\|_2 = 0.71177E-01$$

$$RES_l=\sum\|\varepsilon_{li}\|_2 = 0.24853E+03$$

Mode	$\|\varepsilon_{ri}\|$		$\|\varepsilon_{li}\|$	
	2-Norm	∞-Norm	2-Norm	∞norm
ShPer	0.69848E-02	0.63638E-02	0.66958E+02	0.47344E+02
ShPer	0.69848E-02	0.63638E-02	0.66958E+02	0.47344E+02
P11D	0.59112E-02	0.53929E-02	0.75575E+02	0.53439E+02
DRoll	0.80799E-02	0.75772E-02	0.18091E-01	0.70056E-13
DRoll	0.80799E-02	0.75772E-02	0.18091E-01	0.70056E-13
FPhug	0.38560E-02	0.35073E-02	0.21993E+02	0.15551E+02
Y3D	0.43562E-02	0.39647E-02	0.13785E+02	0.97474E+01
Roll	0.84017E-02	0.74468E-02	0.20977E+00	0.17306E-12
SPhug	0.58469E-03	0.53156E-03	0.20722E+01	0.12600E+01
Filtr	0.12966E-03	0.11710E-03	0.41343E-01	0.11079E-14
Filtr	0.29424E-02	0.27095E-02	0.23058E-01	0.14597E-14
Filtr	0.14411E-01	0.13104E-01	0.69341E+00	0.12940E-05
Filtr	0.28594E-11	0.20895E-11	0.54636E-01	0.33852E-15
Filtr	0.42821E-12	0.35023E-12	0.23159E-01	0.56116E-08
Filtr	0.45401E-03	0.45067E-03	0.10299E+00	0.65734E-15
Spira	0.17388E-09	0.15661E-09	0.00000E+00	0.00000E+00
Filtr	0.50470E-12	0.38527E-12	0.00000E+00	0.00000E+00
Filtr	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00

TABLE 20

RECONFIGURED FEEDBACK GAINS FOR CASE B DAMAGE

	Q	NZ	AA	R
RST	0.214E+00	-0.131E+00	0.256E-03	0.000E+00
LST	0.214E+00	-0.131E+00	0.256E-03	0.000E+00
RLE	-0.480E-02	0.993E-02	0.285E-01	0.000E+00
LLE	-0.480E-02	0.993E-02	0.285E-01	0.000E+00
RTE	0.116E-01	-0.179E-01	-0.852E-03	0.000E+00
LTE	0.116E-01	-0.179E-01	-0.843E-03	0.000E+00
RA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RR	0.000E+00	0.000E+00	0.000E+00	0.689E+00
LR	0.000E+00	0.000E+00	0.000E+00	0.689E+00

	P	NY	C1	C2
RST	0.240E-01	0.000E+00	-0.258E-02	0.748E-03
LST	-0.240E-01	0.000E+00	-0.258E-02	0.748E-03
RLE	0.000E+00	0.000E+00	-0.664E-03	0.000E+00
LLE	0.000E+00	0.000E+00	-0.664E-03	0.000E+00
RTE	0.192E-01	-0.240E-03	0.978E-03	0.000E+00
LTE	-0.192E-01	-0.141E-03	0.978E-03	0.000E+00
RA	0.600E-01	0.000E+00	0.000E+00	0.000E+00
LA	-0.600E-01	0.000E+00	0.000E+00	0.000E+00
RR	0.388E+00	0.165E+02	0.000E+00	0.000E+00
LR	0.388E+00	0.165E+02	0.000E+00	0.000E+00

	C3	C4	C5	C6
RST	-0.192E-01	-0.211E+00	-0.384E-01	0.000E+00
LST	-0.192E-01	-0.211E+00	-0.384E-01	0.000E+00
RLE	-0.494E-02	0.126E-01	-0.988E-02	0.413E-01
LLE	-0.494E-02	0.126E-01	-0.988E-02	0.413E-01
RTE	0.727E-02	-0.235E-01	0.145E-01	-0.112E-03
LTE	0.727E-02	-0.235E-01	0.145E-01	-0.111E-03
RA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RR	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LR	0.000E+00	0.000E+00	0.000E+00	0.000E+00

	C7	C8	C9	C10
RST	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LST	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RLE	0.589E-03	0.000E+00	0.000E+00	0.000E+00
LLE	0.589E-03	0.000E+00	0.000E+00	0.000E+00
RTE	0.210E-01	0.000E+00	0.000E+00	0.000E+00
LTE	0.210E-01	0.000E+00	0.000E+00	0.000E+00
RA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RR	0.000E+00	-0.856E-02	-0.494E-02	0.000E+00
LR	0.000E+00	-0.856E-02	-0.494E-02	0.000E+00

	C11	C12
RST	0.000E+00	0.000E+00
LST	0.000E+00	0.000E+00
RLE	0.000E+00	0.000E+00
LLE	0.000E+00	0.000E+00
RTE	0.000E+00	0.000E+00
LTE	0.000E+00	0.000E+00
RA	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00
RR	0.317E-02	0.000E+00
LR	0.318E-02	0.000E+00

TABLE 21
CASE B RECONFIGURED AIRCRAFT EIGENSTRUCTURE

Spiral		*Phugoid*		*Phugoid*	
λ = -0.0029		λ = -0.0202 + 0.0000J		λ = -0.3699 + 0.0000J	
0.0298	0.0000J	-1.0000	0.0000J	0.6586	0.0000J
-0.0017	0.0000J	0.0535	0.0000J	-1.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	-0.0019	0.0000J
0.0000	0.0000J	-0.0001	0.0000J	0.0051	0.0000J
-0.7768	0.0000J	0.0000	0.0000J	0.0002	0.0000J
-0.0494	0.0000J	0.0000	0.0000J	0.0000	0.0000J
-0.0029	0.0000J	0.0000	0.0000J	0.0000	0.0000J
-1.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J

Roll		*Dutch Roll*		*Dutch Roll*	
λ = -2.8796		λ = -1.8682-2.6152J		λ = -1.8682 + 2.6152J	
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
1.0000	0.0000J	-1.0000	1.2658J	-1.0000	-1.2658J
0.0037	0.0000J	-0.0079	-0.0008J	-0.0017	0.0008J
0.0472	0.0000J	0.0152	-0.0085J	0.0090	0.0085J
-0.0164	0.0000J	-0.0006	0.0054J	-0.0033	-0.0054J

Short Period		*Short Period*	
λ = -1.5052-3.0134J		λ = -1.5052 + 3.0134J	
-0.0085	-0.0096J	-0.0085	0.0096J
-1.0000	0.8656J	-1.0000	-0.8656J
0.0046	0.0041J	0.0065	-0.0041J
-0.0017	0.0007J	-0.0018	-0.0007J
-0.0004	0.0000J	-0.0004	0.0000J
0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J

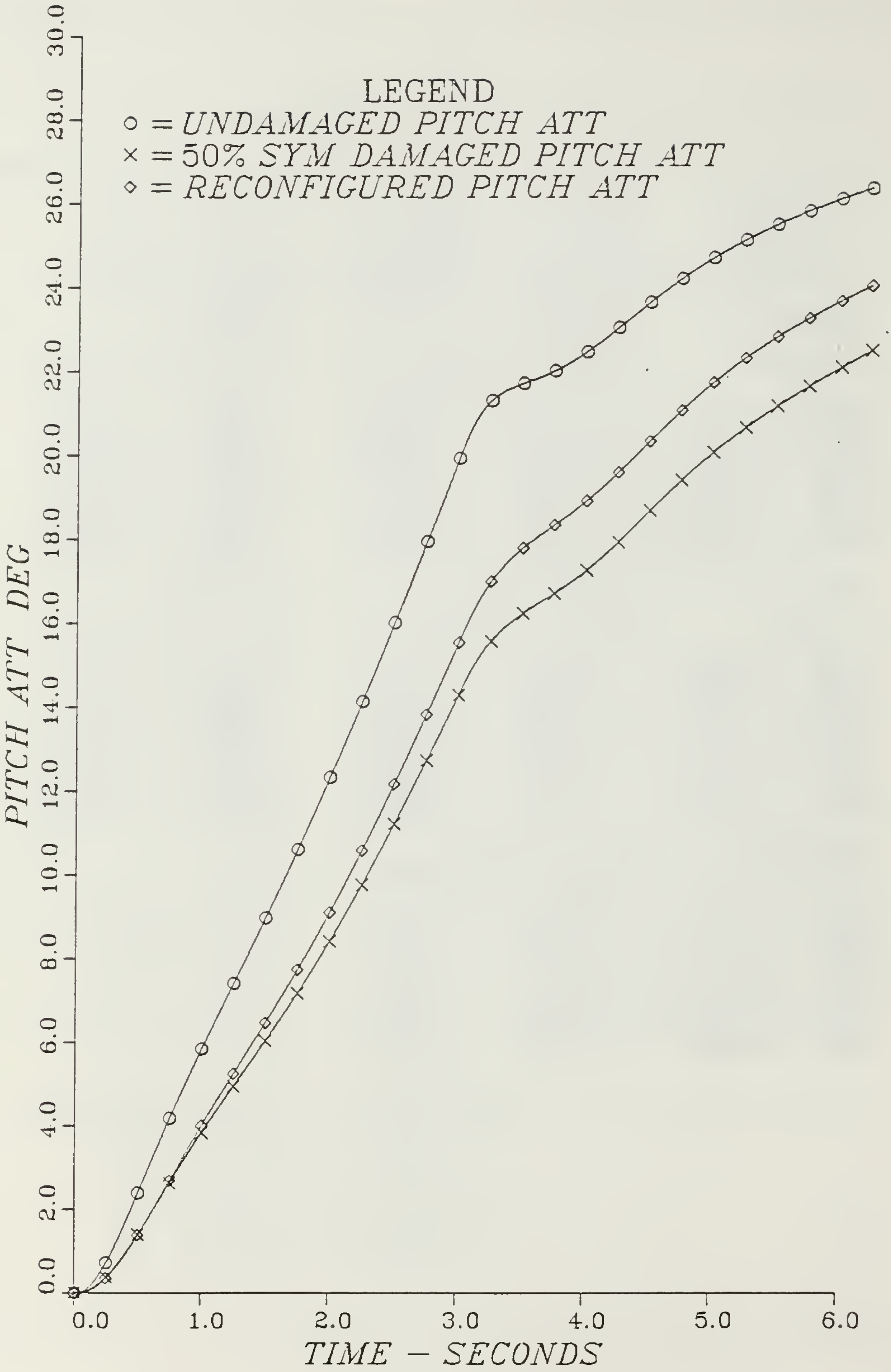


Figure 8.12 Case B Reconfigured Pitch Attitude Response.

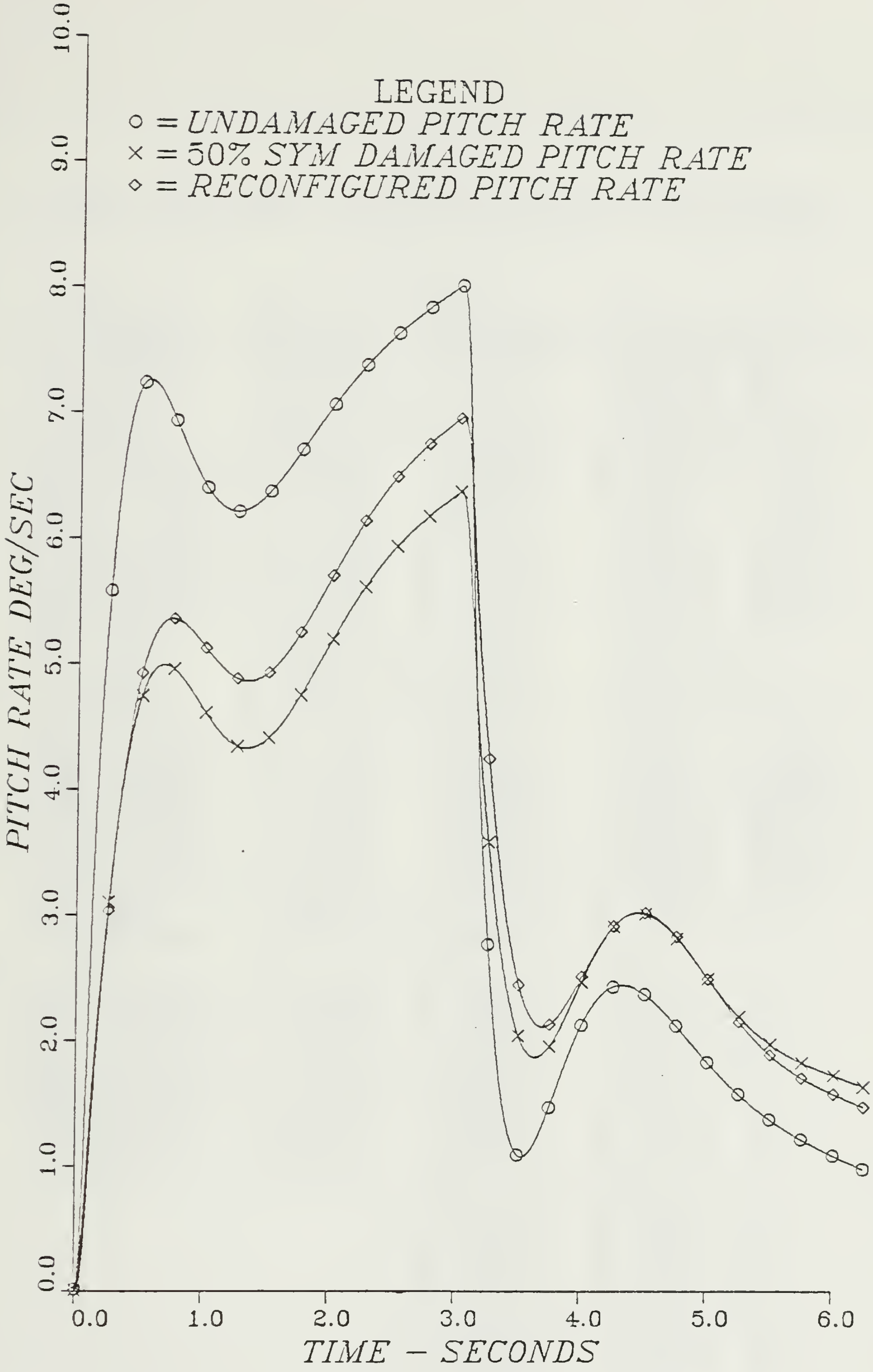


Figure 8.13 Case B Reconfigured Pitch Rate Response.

TABLE 22

PERFORMANCE RECOVERY FEEDBACK GAINS FOR CASE B DAMAGE

	Q	*NZ*	*AA*	*R*
RST	0.213E+00	-0.244E+00	0.184E-02	0.000E+00
LST	0.213E+00	-0.244E+00	0.184E-02	0.000E+00
RLE	-0.175E-01	-0.148E+01	0.501E-01	0.000E+00
LLE	-0.175E-01	-0.148E+01	0.501E-01	0.000E+00
RTE	0.204E-01	0.127E+01	0.618E-02	0.000E+00
LTE	0.204E-01	0.127E+01	0.619E-02	0.000E+00
RA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RR	0.000E+00	0.157E-03	0.000E+00	0.689E+00
LR	0.000E+00	0.105E-03	0.000E+00	0.689E+00
	P	*NY*	*C1*	*C2*
RST	0.240E-01	0.000E+00	-0.271E-02	0.744E-03
LST	-0.240E-01	0.000E+00	-0.271E-02	0.744E-03
RLE	0.000E+00	0.000E+00	-0.360E-02	0.000E+00
LLE	0.000E+00	0.000E+00	-0.360E-02	0.000E+00
RTE	0.192E-01	0.000E+00	0.222E-02	0.000E+00
LTE	-0.192E-01	0.000E+00	0.222E-02	0.000E+00
RA	0.600E-01	0.000E+00	0.000E+00	0.000E+00
LA	-0.600E-01	0.000E+00	0.000E+00	0.000E+00
RR	0.388E+00	0.165E+02	0.000E+00	0.000E+00
LR	0.388E+00	0.165E+02	0.000E+00	0.000E+00
	C3	*C4*	*C5*	*C6*
RST	-0.202E-01	-0.208E+00	-0.403E-01	0.000E+00
LST	-0.202E-01	-0.208E+00	-0.403E-01	0.000E+00
RLE	-0.267E-01	0.399E-01	-0.535E-01	0.418E-01
LLE	-0.267E-01	0.399E-01	-0.535E-01	0.418E-01
RTE	0.205E-01	-0.453E-01	0.330E-01	0.186E-02
LTE	0.205E-01	-0.453E-01	0.330E-01	0.186E-02
RA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RR	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LR	0.000E+00	0.000E+00	0.000E+00	0.000E+00
	C7	*C8*	*C9*	*C10*
RST	0.145E-03	0.000E+00	0.000E+00	0.000E+00
LST	0.145E-03	0.000E+00	0.000E+00	0.000E+00
RLE	0.233E-02	0.000E+00	0.000E+00	0.000E+00
LLE	0.233E-02	0.000E+00	0.000E+00	0.000E+00
RTE	0.205E-01	0.000E+00	0.000E+00	0.000E+00
LTE	0.205E-01	0.000E+00	0.000E+00	0.000E+00
RA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RR	0.000E+00	-0.856E-02	-0.494E-02	0.000E+00
LR	0.000E+00	-0.856E-02	-0.494E-02	0.000E+00
	C11	*C12*		
RST	0.000E+00	0.000E+00		
LST	0.000E+00	0.000E+00		
RLE	0.000E+00	0.000E+00		
LLE	0.000E+00	0.000E+00		
RTE	0.000E+00	0.000E+00		
LTE	0.000E+00	0.000E+00		
RA	0.000E+00	0.000E+00		
LA	0.000E+00	0.000E+00		
RR	0.317E-02	0.000E+00		
LR	0.318E-02	0.000E+00		

TABLE 23

CASE B PERFORMANCE RECOVERY AIRCRAFT EIGENSTRUCTURE

Spiral		*Phugoid*		*Phugoid*	
λ = -0.0029		λ = -0.0182 + 0.0000J		λ = -0.4301 + 0.0000J	
0.0002	0.0000J	-1.0000	0.0000J	0.5162	0.0000J
0.0000	0.0000J	0.0490	0.0000J	-1.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	-0.0019	0.0000J
0.0000	0.0000J	-0.0000	0.0000J	0.0045	0.0000J
0.7394	0.0000J	0.0000	0.0000J	0.0003	0.0000J
0.0494	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
1.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J

Roll		*Dutch Roll*		*Dutch Roll*	
λ = -2.8797		λ = -1.8682-2.6152J		λ = -1.8682 + 2.6152J	
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J	0.0000	0.0000J
-1.0000	0.0000J	-1.0000	-2.4534J	-1.0000	2.4534J
-0.0037	0.0000J	0.0076	-0.0107J	0.0076	0.0107J
-0.0472	0.0000J	-0.0002	0.0286J	-0.0002	-0.0286J
0.0164	0.0000J	-0.0072	-0.0052J	-0.0072	0.0052J

Short Period		*Short Period*	
λ = -1.4342-3.1480J		λ = -1.4342 + 3.1480J	
-0.0159	0.0082J	-0.0159	-0.0082J
-1.0000	-1.7481J	-1.0000	1.7481J
0.0092	-0.0038J	0.0092	0.0038J
-0.0021	-0.0020J	-0.0021	0.0020J
-0.0004	0.0000J	-0.0004	0.0000J
0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J
0.0000	0.0000J	0.0000	0.0000J

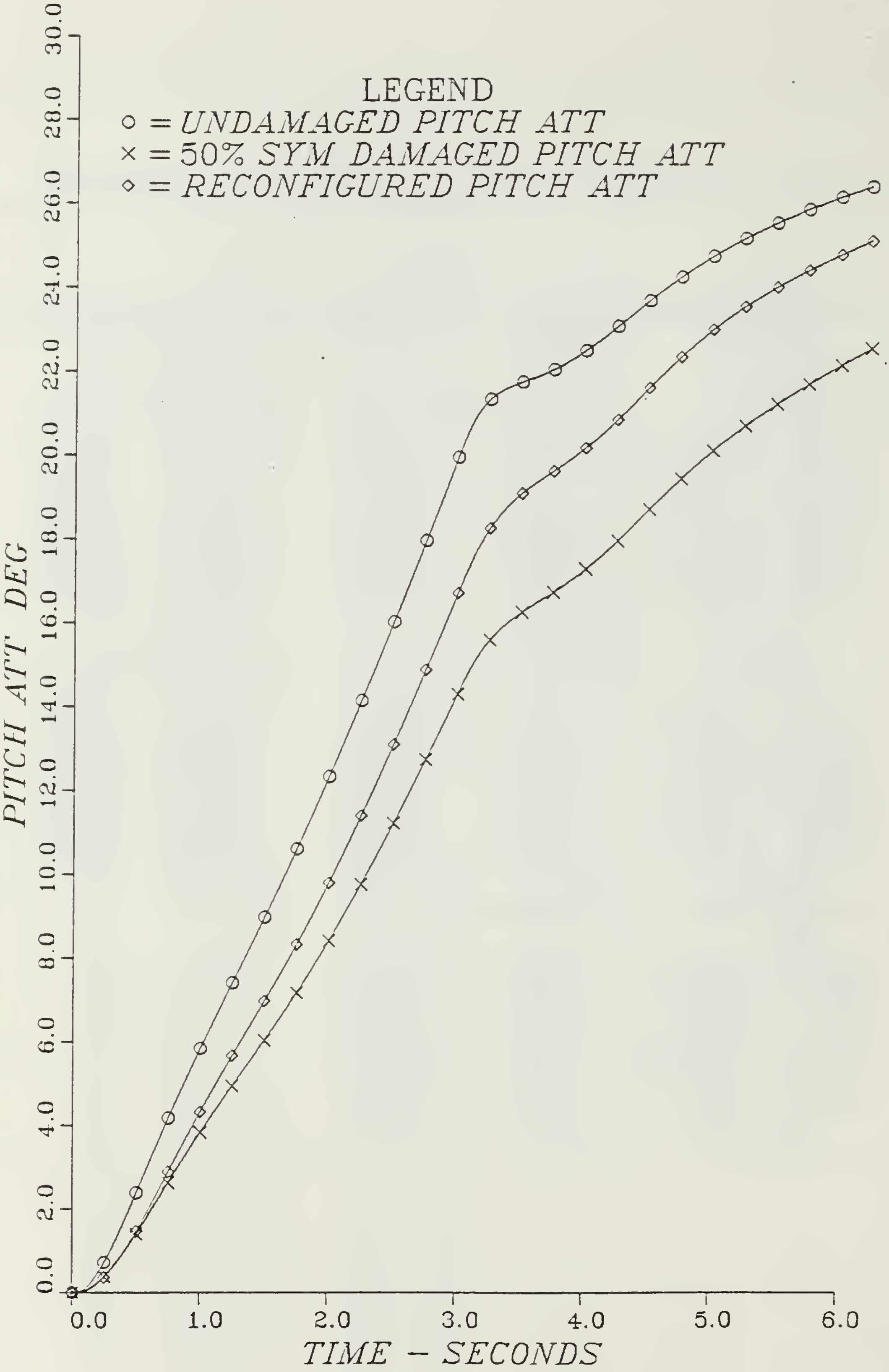


Figure 8.14 Case B Performance Recovery Pitch Attitude Response.

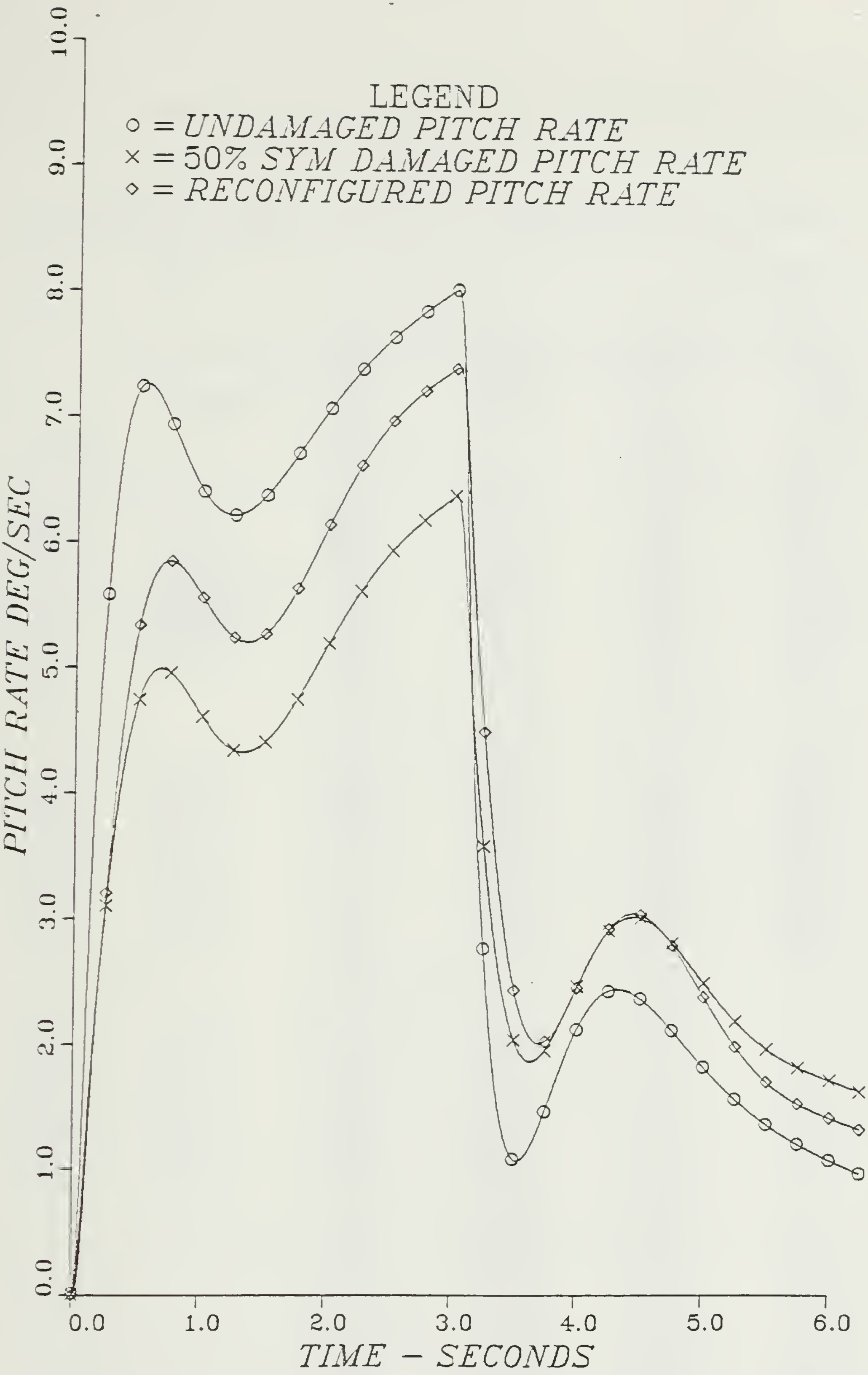


Figure 8.15 Case B Performance Recovery Pitch Rate Response.

TABLE 24

CASE C(25% ASYMMETRIC) DAMAGED C-LOOP EIGENVALUES

Z-Plane Roots		*S-Plane Roots*		*Mode*
1.0000	0.0000J	0.000	0.000J	
1.0000	0.0000J	0.000	0.000J	
1.0000	*0.0000J*	*0.003*	*0.000J*	*Spiral*
1.0000	*0.0000J*	*0.004*	*0.000J*	
0.9998	*0.0000J*	*-0.018*	*0.000J*	*Phugoid*
0.9942	*0.0000J*	*-0.464*	*0.000J*	*Phugoid*
0.9876	0.0000J	-1.000	0.000J	
0.9860	0.0000J	-1.129	0.000J	
0.9821	0.0000J	-1.443	0.000J	
0.9753	0.0000J	-2.000	0.000J	
0.9753	0.0000J	-2.000	0.000J	
0.9753	0.0000J	-2.004	0.000J	
0.9644	*0.0000J*	*-2.898*	*0.000J*	*Roll*
0.9764	*-0.0319J*	*-1.870*	*-2.616J*	*Dutch Roll*
0.9764	*0.0319J*	*-1.870*	*2.616J*	*Dutch Roll*
0.9736	*0.0347J*	*-2.092*	*2.852J*	*Short Period*
0.9736	*-0.0347J*	*-2.092*	*-2.852J*	*Short Period*
0.9558	0.0000J	-3.613	0.000J	
0.8409	0.0000J	-13.861	0.000J	
0.7567	0.0000J	-22.305	0.000J	
0.7109	0.0000J	-27.292	0.000J	
0.7106	0.0000J	-27.329	0.000J	
0.6988	-0.2220J	-24.821	-24.612J	
0.6988	0.2220J	-24.821	24.612J	
0.6985	-0.2223J	-24.849	-24.651J	
0.6985	0.2223J	-24.849	24.651J	
0.7651	-0.3216J	-14.912	-31.832J	
0.7651	0.3216J	-14.912	31.832J	
0.7597	-0.3344J	-14.906	-33.172J	
0.7597	0.3344J	-14.906	33.172J	
0.4907	-0.0613J	-56.336	-9.940J	
0.4907	0.0613J	-56.336	9.940J	
0.4119	0.0000J	-70.951	0.000J	
0.4267	-0.3259J	-49.749	-52.187J	
0.4267	0.3259J	-49.749	52.187J	
0.4191	0.3947J	-44.173	60.434J	
0.4191	-0.3947J	-44.173	-60.434J	
0.4181	-0.3950J	-44.250	-60.555J	
0.4181	0.3950J	-44.250	60.555J	
0.3601	0.0000J	-81.707	0.000J	
0.3601	0.0000J	-81.708	0.000J	
0.2253	0.4029J	-61.847	84.866J	
0.2253	-0.4029J	-61.847	-84.866J	
0.2238	0.4019J	-62.116	85.017J	
0.2238	-0.4019J	-62.116	-85.017J	
0.1226	0.0000J	-167.883	0.000J	
0.1217	0.0000J	-168.484	0.000J	
0.1208	0.0000J	-169.101	0.000J	
-0.0042	0.0203J	-309.929	141.910J	
-0.0042	-0.0203J	-309.929	-141.910J	
0.0017	0.0083J	-381.951	109.379J	
0.0017	-0.0083J	-381.951	-109.379J	
0.0033	0.0000J	-458.310	0.000J	
0.0032	0.0000J	-460.648	0.000J	
0.0031	0.0000J	-461.236	0.000J	

TABLE 25

CASE C (25% ASYMMETRIC DAMAGE) AIRCRAFT EIGENSTRUCTURE

Spiral

$\lambda = 0.0029 + 0.0000J$

-0.0222	0.0000J
0.0012	0.0000J
0.0000	0.0000J
0.0000	0.0000J
-0.7765	0.0000J
-0.0494	0.0000J
-0.0029	0.0000J
-1.0000	0.0000J

Phugoid

$\lambda = -0.0179 + 0.0000J$

-1.0000	0.0000J
0.0561	0.0000J
0.0000	0.0000J
0.0000	0.0000J
0.0016	0.0000J
0.0001	0.0000J
0.0000	0.0000J
0.0024	0.0000J

Phugoid

$\lambda = -0.4641 + 0.000J$

-0.4041	0.0000
1.0000	0.0000
0.0016	0.0000
-0.0034	0.0000
-0.0030	0.0000
0.0001	0.0000
-0.0010	0.0000
0.0021	0.0000

Roll

$\lambda = -2.8978 + 0.000J$

0.0002	0.0000J
-0.0114	0.0000J
0.0000	0.0000J
0.0000	0.0000J
1.0000	0.0000J
0.0037	0.0000J
0.0477	0.0000J
-0.0165	0.0000J

Dutch Roll

$\lambda = -1.8697 - 2.6156J$

-0.0001	-0.0006J
-0.0526	0.0220J
0.0002	0.0002J
-0.0001	0.0000J
1.0000	-2.3749J
0.0126	-0.0021J
-0.0196	0.0197J
-0.0014	-0.0085J

Dutch Roll

$\lambda = -1.8697 + 2.6156$

-0.0001	0.0006
-0.0526	-0.0220
0.0002	-0.0002
-0.0001	0.0000
1.0000	2.3749
0.0126	0.0021
-0.0196	-0.0197
-0.0014	0.0085

Short Period

$\lambda = -2.0921 + 2.8524J$

-0.0027	-0.0087J
1.0000	-0.0450J
-0.0015	0.0045J
0.0013	-0.0004J
-0.0491	0.0123J
-0.0001	0.0002J
0.0012	0.0000J
-0.0002	-0.0003J

Short Period

$\lambda = -2.0921 - 2.8524J$

-0.0027	0.0087J
1.0000	0.0450J
-0.0015	-0.0045J
0.0013	0.0004J
-0.0491	-0.0123J
-0.0001	-0.0002J
0.0012	0.0000J
-0.0002	0.0003J

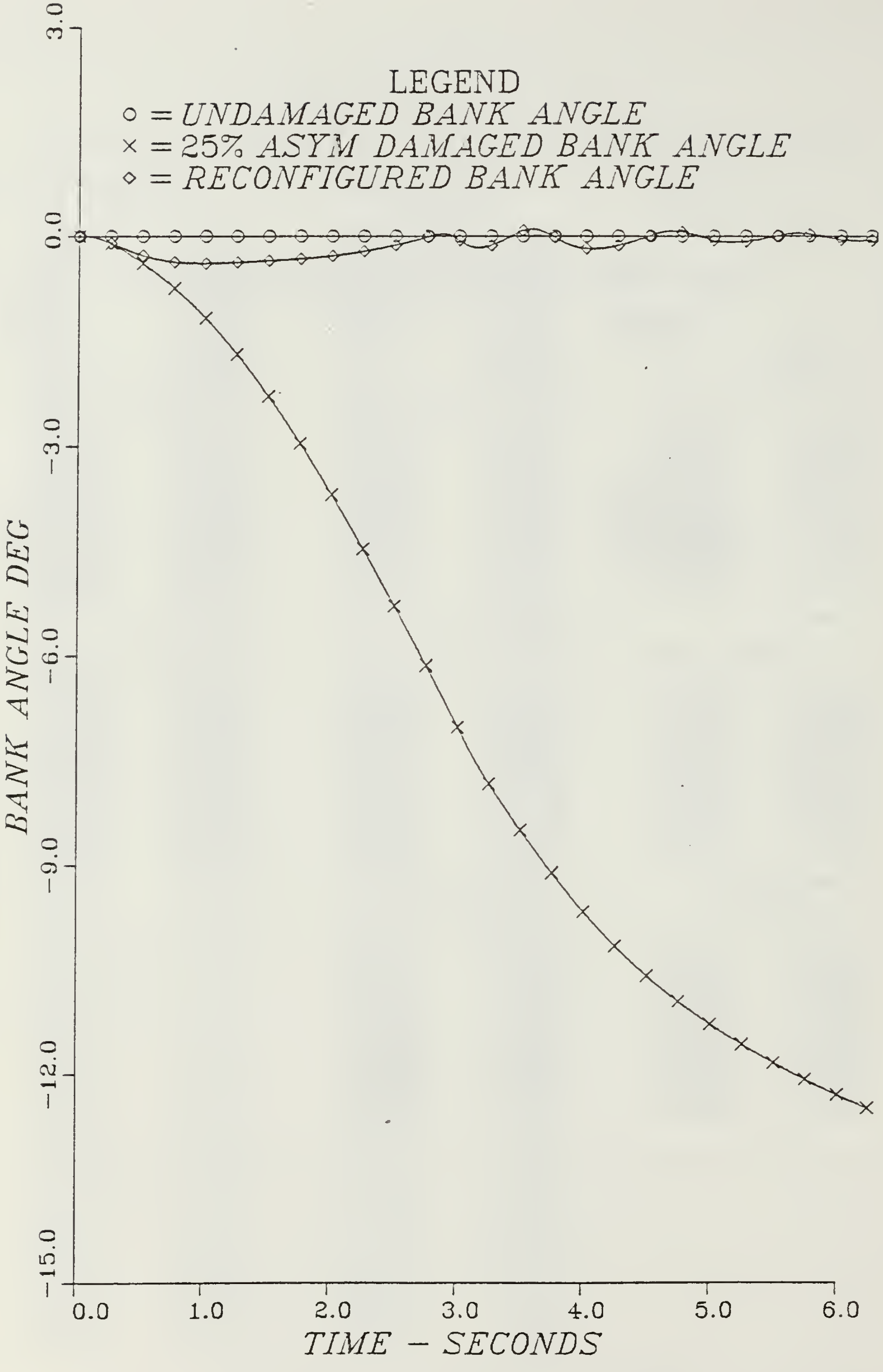


Figure 8.16 Case C Bank Angle Responses.

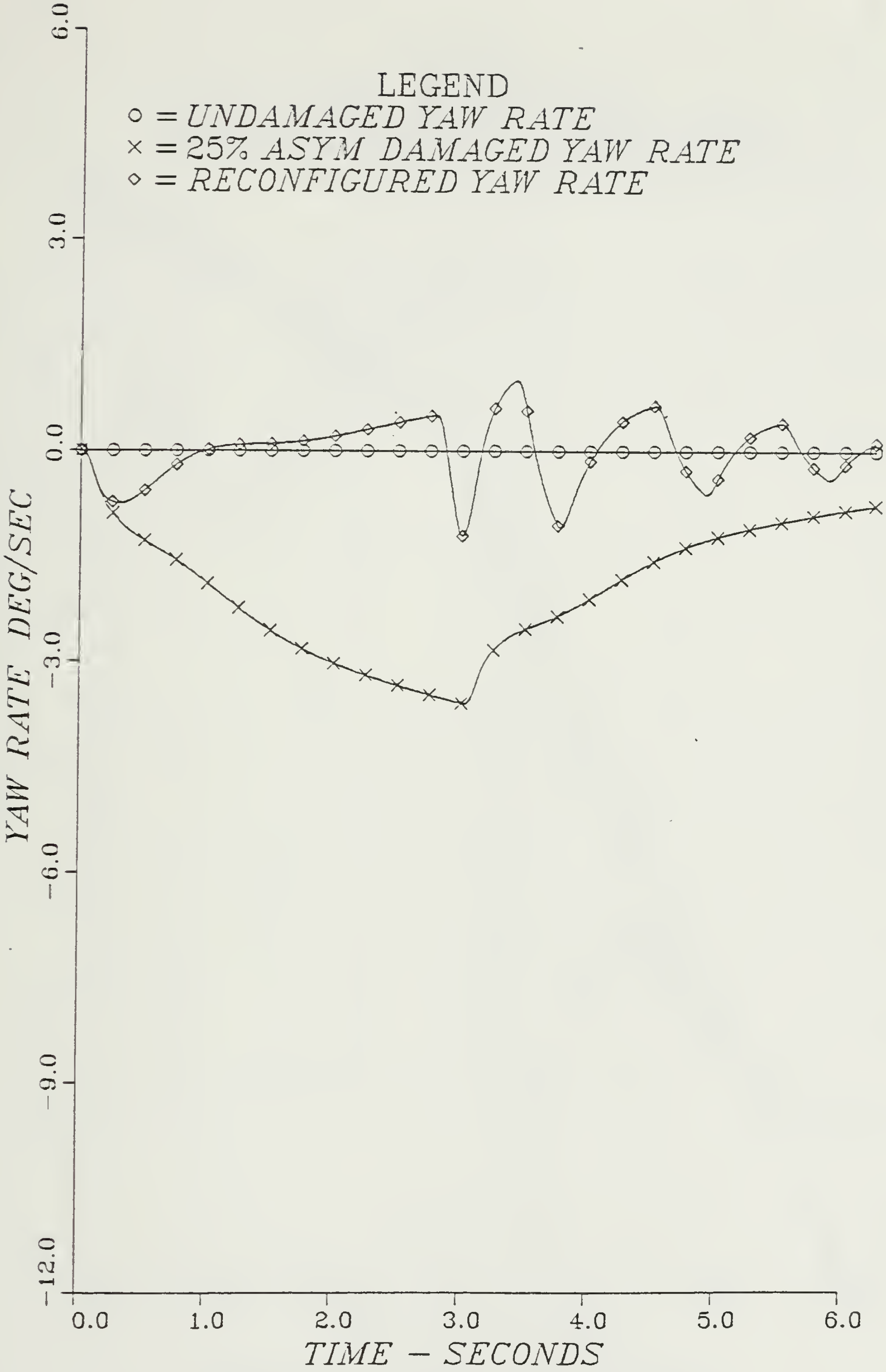


Figure 8.17 Case C Yaw Rate Responses.

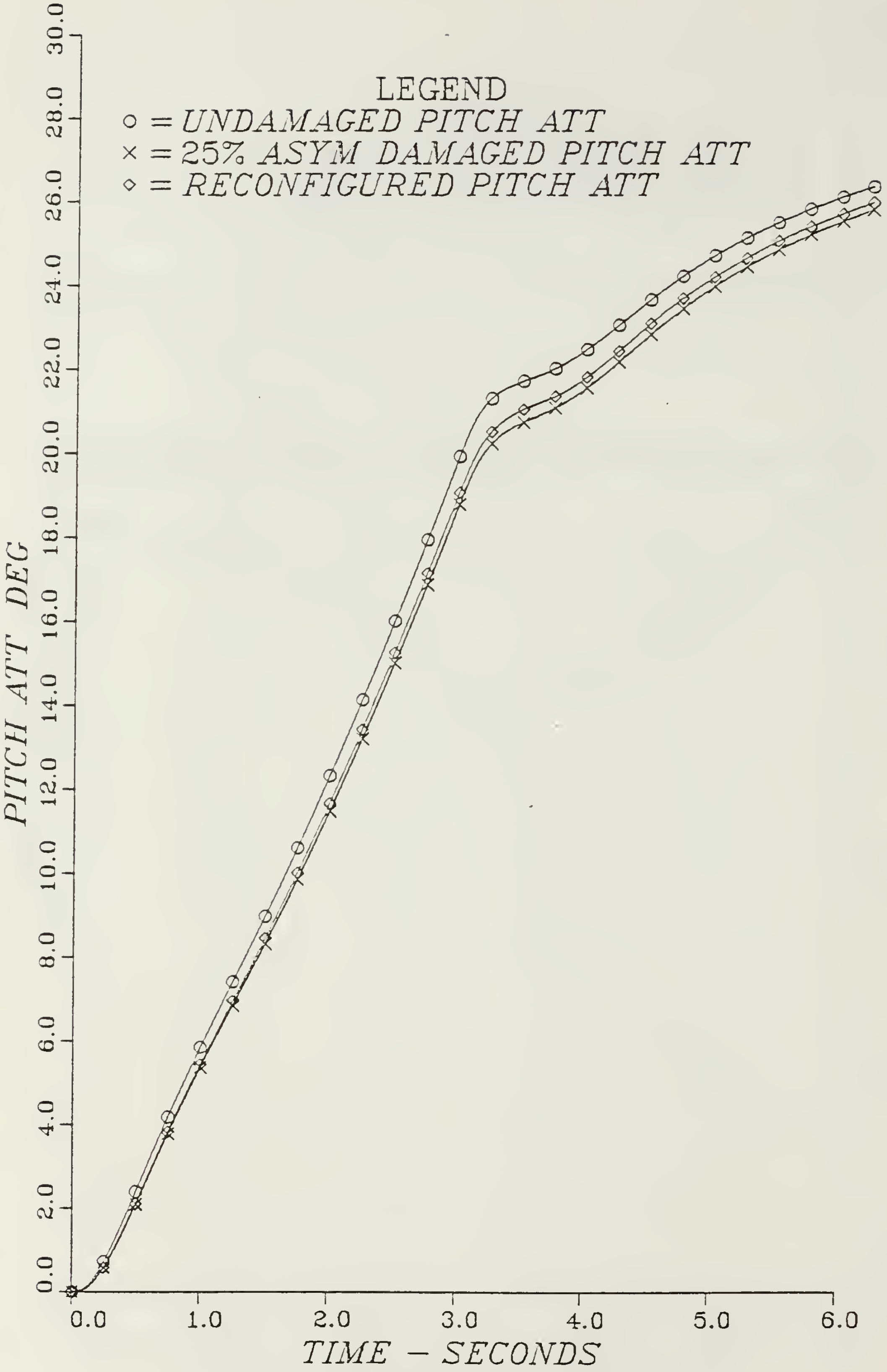


Figure 8.18 Case C Pitch Attitude Responses.

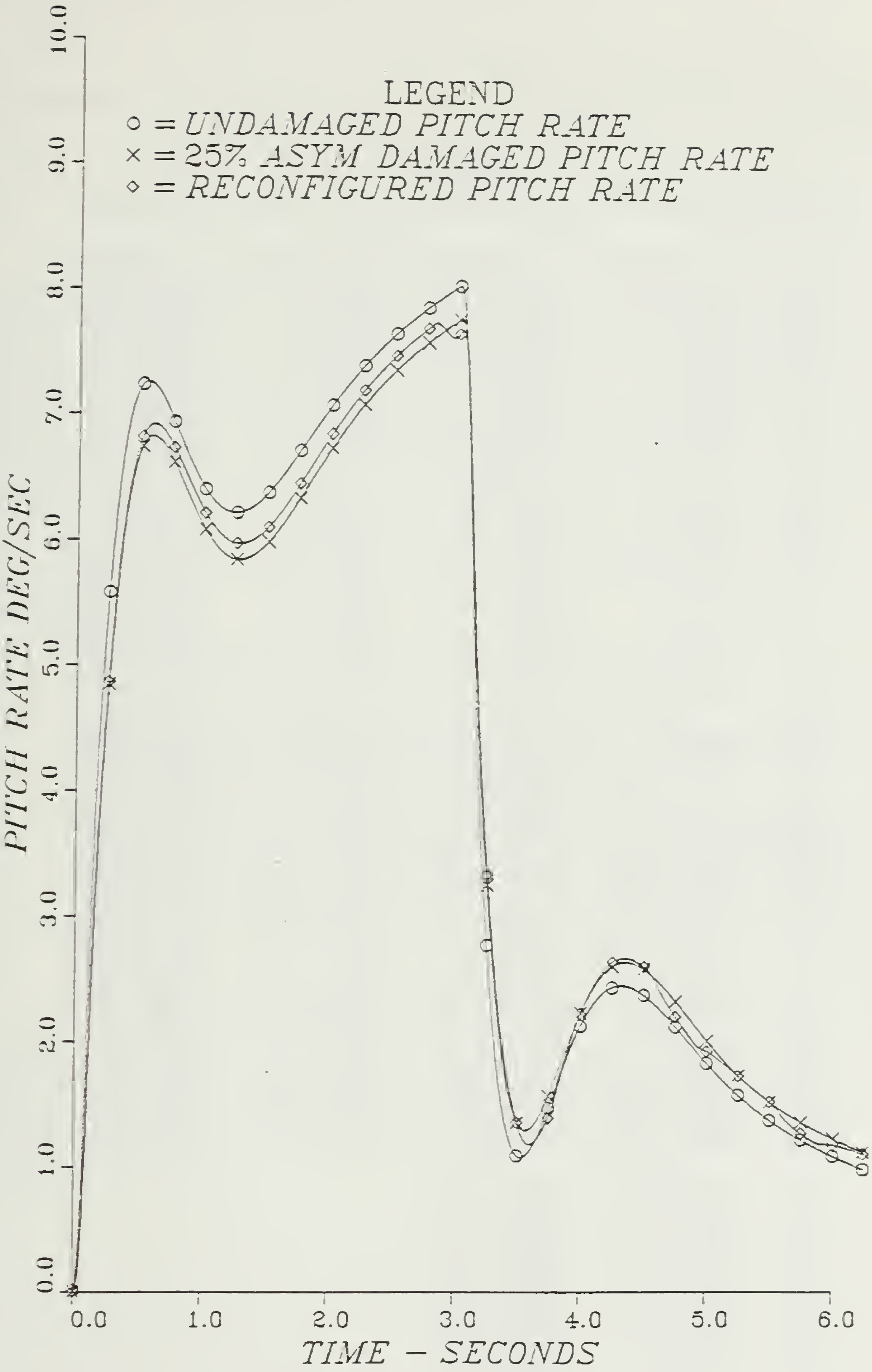


Figure 8.19 Case C Pitch Rate Responses.

TABLE 26

RECONFIGURED FEEDBACK GAINS FOR CASE C DAMAGE

	Q	*NZ*	*AA*	*R*
RST	0.214E+00	-0.131E+00	0.000E+00	0.000E+00
LST	0.214E+00	-0.131E+00	-0.323E-03	0.000E+00
RLE	0.000E+00	0.000E+00	0.209E-01	0.000E+00
LLE	-0.209E-02	0.214E-02	-0.320E-02	0.000E+00
RTE	0.000E+00	0.000E+00	0.110E-01	0.000E+00
LTE	0.394E-02	-0.208E-02	0.427E-01	0.000E+00
RA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RR	0.000E+00	0.000E+00	0.000E+00	0.689E+00
LR	0.000E+00	0.000E+00	0.000E+00	0.689E+00

	P	*NY*	*C1*	*C2*
RST	0.240E-01	0.000E+00	-0.257E-02	0.748E-03
LST	-0.240E-01	0.000E+00	-0.259E-02	0.748E-03
RLE	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LLE	0.000E+00	0.000E+00	-0.364E-03	0.000E+00
RTE	0.192E-01	0.000E+00	0.000E+00	0.000E+00
LTE	-0.192E-01	0.000E+00	0.322E-03	0.000E+00
RA	0.600E-01	0.000E+00	0.000E+00	0.000E+00
LA	-0.600E-01	0.000E+00	0.000E+00	0.000E+00
RR	0.388E+00	0.165E+02	0.000E+00	0.000E+00
LR	0.388E+00	0.165E+02	0.000E+00	0.000E+00

	C3	*C4*	*C5*	*C6*
RST	-0.191E-01	-0.211E+00	-0.383E-01	0.000E+00
LST	-0.193E-01	-0.211E+00	-0.385E-01	0.000E+00
RLE	0.000E+00	0.000E+00	0.000E+00	0.412E-01
LLE	-0.271E-02	0.431E-02	-0.541E-02	0.403E-01
RTE	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LTE	0.240E-02	-0.523E-02	0.479E-02	0.210E-02
RA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RR	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LR	0.000E+00	0.000E+00	0.000E+00	0.000E+00

	C7	*C8*	*C9*	*C10*
RST	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LST	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RLE	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LLE	-0.252E-02	0.000E+00	0.000E+00	0.000E+00
RTE	0.219E-01	0.000E+00	0.000E+00	0.000E+00
LTE	0.241E-01	0.000E+00	0.000E+00	0.000E+00
RA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00	0.000E+00	0.000E+00
RR	0.000E+00	-0.856E-02	-0.494E-02	0.000E+00
LR	0.000E+00	-0.856E-02	-0.494E-02	0.000E+00

	C11	*C12*
RST	0.000E+00	0.000E+00
LST	0.000E+00	0.000E+00
RLE	0.000E+00	0.000E+00
LLE	0.000E+00	0.000E+00
RTE	0.000E+00	0.000E+00
LTE	0.000E+00	0.000E+00
RA	0.000E+00	0.000E+00
LA	0.000E+00	0.000E+00
RR	0.317E-02	0.000E+00
LR	0.318E-02	0.000E+00

IX. CONCLUSIONS AND RECOMMENDATIONS

Applications of eigenstructure assignment to design of robust decoupling and reconfiguring controllers has been shown to be an alternative tool for the control system designer. A level of sophistication over conventional design techniques is required due to the computational problems which arise from high order systems. Specifically, the majority of the computational complexities arise from the vast degrees of freedom for eigenvector orientation available in hyperspace. As the designer must clearly be aware of the modal structure of the system at hand, the technique is not to be envisioned as a black box design algorithm. The designer must become an active participant during all steps of the design. Further research is recommended in the following areas:

a. Further investigation into obtaining performance recovery via modified objective functions for the reconfiguring controller problem is warranted.

b. Developing more efficient optimization schemes for convergence to reoriented eigenvectors. This will allow the researcher to more actively engage in the study of the theoretical aspects of eigenstructure assignment by relieving the burden of computational time required for such convergence.

c. Additional research into analyzing the solution to the asymmetric degradation problem is required for refinement of the technique. Research objectives associated with this type of damage would also involve modeling the cross coupling of stability and control derivatives for several coupling scenarios.

d. Application of eigenstructure assignment to design of robust observers and reconfiguring damaged observers is a natural extension of the work reported in this thesis.

In summary, applications of eigenstructure assignment to damaged controllers has been shown to provide solutions to a specific class of aircraft damage. Solution techniques to the asymmetric and symmetric degradation problems must be further refined before progressing to the combined actuator, sensor, control surface degradation scenarios.

APPENDIX A

NOTATION

n	No. of states
m	No. of inputs
l	No. of outputs
c	No. of commands
A	Plant Matrix (nxn)
B	Control Matrix (nxm)
C	Output Matrix ($l$xn)
D	Feedforward-Output Matrix ($l$xm)
F	Feedback Gain Matrix (mx$l$)
G_1	Feedfwd Command-State Matrix (nxc)
G_2	Feedfwd Command-Input Matrix (mxc)
X	Right Eigenvector Matrix (nxn)
Δ	Diag matrix of c-loop eigenvalues
T	Left Eigenvector Matrix (nxn)
I_n	Identity Matrix (nxn)
Δ	Closed Loop Eigenvalue Matrix (nxn)
A^T	Transpose of matrix A
A^{-1}	Inverse of matrix A
A^+	Pseudo-Inverse of matrix A
SVD(A)	Singular Value Decomposition of matrixA $= U_a \Sigma_a V_a{}^T$
Σ	Diag matrix of singular values
x_i	ith closed loop right eigenvector
$\dot{x}$	dx/dt
$x(k)$	discrete state variable
t_i	ith closed loop left eigenvector
y	output vector
u	control input vector
λ_i	ith closed loop eigenvalue
σ_1	Max singular value
σ_{min}	Min singular value of rank(m) matrix
$\|A\|_2$	Matrix Spectal Norm of matrix A

$\|x\|_2$	Euclidean Norm of vector x
$\mathscr{L}$ or $\mathcal{N}$	Null Space
d( )	dimension of ( )
Φ^{nxn}	Null Matrix of order nxn

APPENDIX B

CONSTRAINTS ON EIGENSTRUCTURE SPECIFICATION

In order to show the constraints on the specification of the closed loop eigenstructure, rewrite equation 3.60 in the following way,

$$BFC = X\Delta X^{-1} - A \qquad \text{(eqn B.1)}$$

Post multiplying equation B.1 by X yields,

$$BFCX = X\Delta - AX \qquad \text{(eqn B.2)}$$

Now let us define an 'n x n' transformation matrix, Q^{-1} [Ref. 23], such that B is transformed into a matrix with the 'mxm' identity in the first 'm' rows and an 'n-m x m' null matrix in the bottom 'n - m' rows,

$$Q = \{ B \mid U_{b1} \} \in \mathscr{R}^{nxn} \qquad \text{(eqn B.3)}$$

$$Q \begin{Bmatrix} I_m \\ \Phi \end{Bmatrix} = B \qquad \text{(eqn B.4)}$$

$$\begin{Bmatrix} I_m \\ \Phi \end{Bmatrix} = Q^{-1}B \qquad \text{(eqn B.5)}$$

Further, use the transformation matrix Q to define the following similarity transformations,

$$CQ = C^0 \qquad \text{(eqn B.6)}$$

$$Q^{-1}X = X^0 \qquad \text{(eqn B.7)}$$

$$Q^{-1}AQ = A^0 \qquad \text{(eqn B.8)}$$

Pre multiplying equation B.2 by Q^{-1} yields,

$$Q^{-1}BFCX = Q^{-1}X\Delta - Q^{-1}AX \qquad \text{(eqn B.9)}$$

Noting that $Q^{-1}B$ results in the lead identity matrix discussed previously and substituting the similarity form of X yields,

$$\begin{Bmatrix} I_m \\ \Phi \end{Bmatrix} FCQX^0 = Q^{-1}X\Delta - Q^{-1}AQX^0 \qquad \text{(eqn B.10)}$$

Further substituting the similarity forms for C,X, and A into equation B.10 yields the similarity form of equation B.2,

$$\begin{Bmatrix} I_m \\ \Phi \end{Bmatrix} FC^0X^0 = X^0\Delta - A^0X^0 \qquad \text{(eqn B.11)}$$

In order to simplify the analysis, let us remove the superscript notation for the similarity transformations from equation B.11 . Equation B.11 is then partitioned into upper and lower blocks.

The upper half left hand block of equation B.11 becomes,

$$FCX_{nxl} = X_{mxl}\Delta_l - A_{mxn}X_{nxl} \qquad \text{(eqn B.12)}$$

where the subscripts denote the row and column dimensions of the respective matrices.

Solving for F from equation B.12 yields an exact solution for F,

$$F = (X_{mxl}\Delta_l - A_{mxn}X_{nxl})(CX_{nxl})^{-1} \qquad \text{(eqn B.13)}$$

The lower block of equation B.11 yields the following equation,

$$\Phi_{n-mxl} = X_{n-mxl}\Delta_l - A_{n-mxn}X_{nxl} \qquad \text{(eqn B.14)}$$

Equation B.14 reveals that there are 'n - m' equations with 'n' unknowns for each closed loop eigenvalue λ_i. Each of the 'l' system of equations is therefore underspecified and one can only specify 'm' elements of each column of X_{nxl}. This is of course the classic result of Srinathkumar [Ref. 11] which is stated in Chapter II. Equation B.13 is the result noted in the work by Srinathkumar [Ref. 11] and Sobel and Shapiro [Ref. 23]. The expressions presented in Table I of Chapter III as noted previously are least square solutions for the feedback gains. Note that if one desires to invoke 'n' eigenvalues and 'n' elements of the eigenvector in equation B.13, then equation B.13 also becomes a least square solution due to CX_{nxl} becoming non square.

APPENDIX C

EIGENS

```
C*********************************************************************
C
C          EIGENVALUE/EIGENVECTOR ASSIGNMENT VIA MOORE (1976)
C          AND KAUTSKY,ET.AL. (1985) ALGORITHMS. PRESENT PROGRAM
C          ACCOMODATES LTI SYSTEMS WITH STATE OR OUTPUT FEEDBACK.
C          MOORE ALGORITHMS ONLY ACCOMODATE SYSTEMS WITH REAL,DISTINCT,
C          & CONTROLLABLE EIGENVALUES AND STATE FEED. KAUTSKY ALGORITHM
C          WILL ACCOMODATE SYSTEMS WITH COMPLEX CONTROLLABLE EIGENVALUES
C          AND WILL FIND ACHIEVABLE EIGENVECTORS ASSOCIATED WITH DESIRED
C          C-LOOP EIGENVALUES FOR BOTH STATE AND OUTPUT FEEDBACK.
C          ADDITIONAL OPTIONS INCLUDE USING THE ADS OPTIMIZATION ROUTINE
C          FOR:    1. MINIMIZING THE CONDITION NO. OF THE EIGENVECTOR
C                     MATRIX. (PRESENTLY NOT WORKING)
C                  2. MINIMIZING THE SINGULAR VALUE OF THE RETURN
C                     DIFFERENCE MATRIX WITH THE CONSTRAINTS OF A DE-
C                     SIRED EIGENSTRUCTURE.
C
C                       CDR V.F. GAVITO
C                         APRIL 1986
C
C*********************************************************************
C
C                       DEFINITIONS
C
C*********************************************************************
C
C             A = PLANT MATRIX            (N X N)
C             AT = A TRANSPOSE            (N X N)
C             B = CONTROL MATRIX          (N X M)
C             BDEL = DAMAGED CONTROL MAT (N X M)
C             C = OUTPUT MATRIX           (L X N)
C             D = FEED FORWARD MATRIX (LXM)
C             EIG = O-LOOP EIGENVALUES    (N X 1)  COMPLEX VECTOR
C             EIGO = O-LOOP EIGENVALUES   (N X I)  (REAL PART)
C             EIGOI = O-LOOP      "       (N X I)  (IMAGINARY PART)
C             EIGD = DESIRED C-LOOP EIGENVALUES    (N X 1)  COMPLEX VECT
C             EIGDD = DESIGN C-LOOP EIGENVALUES       "         "     "
C             EIGREC = C-LOOP EIGENVALUES OF RECONFIGURED SYSTEM
C             SLAMBDA = <LAMDHA*I-A  B>
C             RANK1 = R<SLAMBDA>
C             RANK2 = R<C>
C             VD = DESIRED BASE EIGENVECTORS      (N X N)
C             V = DESIGNED BASE EIGENVECTORS      (N X N)
C             VRECON = EIGENVECTORS (NORMALIZED) OF RECONFIGURED SYSTEM
C             CV = <C>*<V>                        (L X N)
C             E =     "   C*V FOR EACH EIGENVALUE (L X N)
C             WC1.2,3,4.5 = WORK COLUMN VECTORS  (1 X N)
C             WM1.2.3.4.5.6 = WORK MATRICES       (N X N)
C             7,8,9.10.11,12=   "     "               "
C             WKI.2,3,4.5 = WORK AREAS FOR EIGRF
C             W = FREQUENCY (RAD/SEC)
C             WP =     "          "     FOR DISSPLA COMPATIBILITY
C             K = I (FOR FORCING EQUALITY RELATION FOR
C                    UNCONTROLLABLE EIGENVALUES)
C             F = FEEDBACK GAIN MATRIX            (M X L)
C             FC = COMPLEX FEEDBACK GAIN MATRIX  (M X L)
C             FDEL = RECONFIGURED FEEDBACK GAINS (M X N)
C             ABF = C-LOOP MATRIX  A + BF
C             ICTR = CONTROLLABILITY FLAG FOR O-LOOP EIGENVALUES
C             IFEED = FEEDBACK FLAG
C                    1 = FULL STATE FEEDBACK
```

```
C                    2 = OUTPUT FEEDBACK
C                    3 = OUTPUT FEEDBACK PLUS FEED FGRWARO
C             SF = SINGULAR VALUE VECTOR FOR FEEDBACK GAINS
C             S1 = SINGULAR VALUE VECTOR 1
C             S2 = SINGULAR VALUE VECTOR 2
C             SVM = MIN SINGULAR VALUE OF I + FG
C             SVM1 = MIN SINGULAR VALUE OF I + GF
C             SVMAX = MAX   "        "   "  I + FG
C             SVMAX1 = "    "        "   "  I + GF
C             SVMP = MAX   "        "   "    "  FG FOR DISSPLA COMPAT
C             SMREC = MIN S.V. CF RECONFIGURED SYSTEM (A SINGLE VALUE)
C             WRED = FREQUENCY OF SMREC
C             UI,VI = ORTHOGONAL MATRICES COMPUTED BY CSVO FOR SVD OF
C                     RETURN DIFFERENCE MATRIX
C             UIF,VIF = ORTHOGCNAL MATRICES COMPUTED BY LSVDF FOR SVD OF
C                       FEEDBACK GAIN MATRIX
C             Z,Z1,Z2,Z3,Z4,Z5 = COMPLEX WORK AREAS
C             UTSLAM = U**TRANSPOSE OF SVD OF SLAMBDA
C             VRT = U**TRANSPOSE OF EIGENVECTOR MATRIX
C
C**********************************************************************
C
C                  DIMENSION STATEMENTS
C
C**********************************************************************
C
            IMPLICIT REAL*8(A-H,P-Z)
            REAL*8 A(10,10),B(10,10),C(10,10),EIGO(20),EIGV(10,10),EIGOI(20),
          1P(10,10),FSAVE(10,10),BDEL(10,10),FDEL(10,10),WM8(10,10),
          2WC1(10),WC3(10),WC4(10),WC5(10),F(10,10),CV(10,10),WM9(10,10),
          3UNITY(10,10),W(2000),SVM1(2000),SVMAX1(2000),WM10(10,10),
          4WM3(10,10),WM5(10,10),UTSLAM(10,10),VRT(10,10),WM11(10,10),
          5WK2(250),ABFDEL(10,10),D(10,10),WAREA2(10,10),WM12(10,10),
          6TEIGR(10),TEIGI(10),TEIGVR(10,10),TEIGVI(10,10),AT(10,10),WK3(20),
          7WM6(10,10),WM7(10,10),SVM(2000),EIGVD(10,10),S3(10),SVMAX(2000)
            DIMENSION COM(10),TITLE(20),WP(2000),SVMP(2000)
            COMPLEX*16Z(10,10),Z1(10),Z2(10,10),Z3(10),Z4(10,10),Z5(10),
          1EIG(10),V(10,10),EIGD(10),EIGDD(10),E(10,10),VO(10,10),
          2 UNITYC(10,10),ZN,VIF(10,10),EIGREC(10),VRECON(10,10),
          3 FC(10,10)
            REAL*8 WM1(10,10),WK1(20),WK4(20),ABF(10,10),SLAMDA(20,20),
          1WM2(10,10),S1(10),WK(20),VR(10,10),S2(10),WC2(10),WKAREA(250),
          2WM4(10,10),WAREA1(250),VSAVE(10,10),VRS(10,10),SF(10),U1F(10,10)
            INTEGER IWK(10),ICTR(10),N,M,L,IFEED
C
C**********************************************************************
C
C             INITIALIZE ALL MATRICES
C
C**********************************************************************
C
            DO 1 I = 1,10
            IWK(I) = 0
            ICTR(I) = 0
            WC1(I) = 0.D0
            WC2(I) = 0.D0
            WC3(I) = 0.D0
            WC4(I) = 0.D0
            WC5(I) = 0.D0
            S1(I) = 0.00
            S2(I) = 0.00
            S3(I) = 0.00
            TEIGR(I) = 0.D0
            TEIGI(1) = 0.D0
            COM(I) = 0.D0
            DO 2 J = 1,10
            A(I,J) = 0.D0
            B(I,J) = 0.D0
            C(I,J) = 0.D0
            D(I,J) = 0.D0
            F(I,J) = 0.D0
            CV(I,J) = 0.D0
            SLAMDA(I,J) = 0.D0
            UTSLAM(I,J) = 0.D0
            VRT(I,J) = 0.D0
            EIGVD(I,J) = 0.D0
            VD(I,J) = (0.D0,0.D0)
            V(I,J) = (0.D0,0.D0)
            VR(I,J) = 0.D0
            VRS(I,J) = (0.D0,0.D0)
            ABF(I,J) = 0.D0
            AT(I,J) = 0.D0
            P(I,J) = 0.D0
            UNITY(I,J) = 0.D0
            UNITYC(I,J) = (0.D0,0.D0)
            UIF(I,J) = 0.D0
            WM1(I,J) = 0.D0
            WM2(I,J) = 0.D0
            WM3(I,J) = 0.D0
            WM4(I,J) = 0.D0
            WM5(I,J) = 0.D0
            WM6(I,J) = 0.D0
            WM7(I,J) = 0.D0
            WM8(I,J) = 0.D0
            WM9(I,J) = 0.D0
            WM10(I,J) = 0.D0
            WM11(I,J) = 0.D0
            WM12(I,J) = 0.D0
2     CONTINUE
1     CONTINUE
            DO 5 I = 1,20
            DO 5 J = 1,20
            SLAMDA(I,J) = 0.D0
5     CONTINUE
            DO 6 I = 1,20
            WK(I) = 0.D0
            WK1(I) = 0.D0
            WK3(I) = 0.D0
            WK4(I) = 0.D0
            EIGO(I) = 0.D0
            EIGOI(I) = 0.D0
6     CONTINUE
            DO 7 I = 1,250
            WK2(I) = 0.D0
            WKAREA(I) = 0.D0
            WAREA1(I) = 0.D0
7     CONTINUE
            DO 8 I = 1,10
            Z1(I) = (0.D0,0.D0)
            Z3(I) = (0.D0,0.D0)
            Z5(I) = (0.D0,0.D0)
            EIG(I) = (0.D0,0.D0)
            EIGD(I) = (0.D0,0.D0)
            EIGDD(I) = (0.D0,0.D0)
            DO 9 J = 1,10
            E(I,J) = (0.D0,0.D0)
            Z(I,J) = (0.D0,0.D0)
            Z2(I,J) = (0.D0,0.D0)
            Z4(I,J) = (0.D0,0.D0)
9     CONTINUE
8      CONTINUE
            DO 11 I = 1,2000
            SVMAX(I) = 0.D0
            SVM(I) = 0.D0
            SVM1(I) = 0.D0
            SVMAXI(I) = 0.D0
            W(I) = 0.D0
            WP(I) = 0.0
            SVMP(I) = 0.0
11    CONTINUE
C
C***********************************************************************
C
C                            READ IN DATA
C
C***********************************************************************
C        READ IN OUTPUT TITLE
C     GOTO 511
12    READ(1,16)TITLE
16    FORMAT(20A4)
C      READ IN NO. OF STATES, NO. OF INPUTS, NO. OF OUTPUTS, TYPE FEEDBA
            READ(1,10) N,M,L,IFEED
10    FORMAT(4I2)
C        READ IN A,B,C,D
            DO 20 I = 1,N
            READ(1,15) (A(I,J),J=1,N)
20    CONTINUE
            DO 30 I = 1,N
            READ (1,15) (B(I,J),J=1,M)
30    CONTINUE
            DO 40 I = 1,L
            READ(1,15) (C(I,J),J=1,N)
40    CONTINUE
            IF(IFEED.NE.3)GOTO 22
21    DO 51 I = 1,L
51    READ(1,15)(D(I,J),J=1,M)
15    FORMAT(6F12.5)
C     READ IN DESIRED C-LOOP EIGENVALUES
22    DO 41 I = 1,N
            READ(1,42)EIGD(I)
42    FORMAT(2F12.5)
41    CONTINUE
C     READ IN DESIRED C * V
            DO 43 I = 1,N
            DO 43 J = 1,L
            READ(1,44)E(J,I)
44    FORMAT(2F12.5)
43    CONTINUE
C     READ IN DESIRED BASE EIGENVECTORS
            DO 45 I = 1,N
            DO 45 J = 1,N
            READ(1,46)VD(J,I)
46    FORMAT(2F12.5)
45    CONTINUE
C***********************************************************************
C
C             DISPLAY INPUT FOR CHECKING
C
C***********************************************************************
            CALL FRTCMS ('CLRSCRN ')
            WRITE(6,140)TITLE
            WRITE(6,52)IFEED
52    FORMAT(/5X,'**** FEEDBACK FLAG = ',I2,' ****',/)
            WRITE(6,50)
50    FORMAT(/10X,'**** A MATRIX ****',/)
            DO 70 I = 1,N
            WRITE(6,60) (A(I,J),J=1,N)
60    FORMAT(1X,6(E12.5,2X))
70    CONTINUE
            WRITE(6,80)
80    FORMAT(/10X,'**** B MATRIX ****',/)
            DO 100 I = 1,N
            WRITE(6,90) (B(I,J),J=1,M)
90    FORMAT(1X,6(E12.5,2X))
100   CONTINUE
            WRITE(6,110)
110   FORMAT(/10X,'**** C MATRIX ****',/)
```

```
          DO 130 I = 1,L
          WRITE(6,120) (C(I,J),J=1,N)
120   FORMAT(1X,6(E12.5,2X))
130   CONTINUE
          WRITE(6,111)
111   FORMAT(/10X,'**** D MATRIX ****',/)
          DO 112 I = 1,L
112   WRITE(6,120)(D(I,J),J=1,M)
          WRITE(6,150)
150   FORMAT(//10X,'**** DESIRED BASE EIGENVECTORS FOR USE IN ****',/
         110X,        '****         KAUTSKY ALGORITHM ONLY         ****'//
         2 2X,'EIGENVECTOR',2X,'REAL PART',6X,'IMAG PART',/)
          DO 151 I = 1,N
          WRITE(6,153)I
153   FORMAT(7X,I2)
          DO 151 J = 1,N
          WRITE(6,152)VD(J,I)
152   FORMAT(13X,E12.5,3X,E12.5)
151   CONTINUE
140   FORMAT(//,20A4)
          WRITE(6,160)
160   FORMAT(/10X,'**** DESIRED C-LOOP EIGENVALUES FOR USE IN ****',/
         110X,        '****          KAUTSKY ALGORITHM ONLY         ****',//2X
         2,'EIGENVALUE',2X,'REAL PART',6X,'IMAG PART',/)
          DO 155 I = 1,N
          WRITE(6,156)I,EIGD(I)
156   FORMAT(7X,I2,4X,E12.5,3X,E12.5)
155   CONTINUE
C**********************************************************************
C
C                 CONSTANTS
C
C**********************************************************************
C
C             IDENTITY MATRIX
          DO 200 I = 1,10
          UIF(I,I) = 1.D0
          UNITY(I,I)=1.D0
          UNITYC(I,I) = (1.D0,0.D0)
200   CONTINUE
C     GOTO 531
C**********************************************************************
C
C       DECIDE ON RECONFIGURATION ANALYSIS OR EIGENSTRUCTURE DESIGN
C
C**********************************************************************
C     WRITE(6,202)
C02   FORMAT(/5X,'**** ENTER "1" FOR RECONFIGURATION STUDIES ****',/5X,
C    1             '**** ENTER "2" FOR EIGENSTRUCTURE ASSIGNMENT **',/)
C     READ(*,313)IRECON
C     IF(IRECON.EQ.2)GOTO 165
C     ISOLN = 3
C     CALL RECON(A,B,C,F,L,M,N,BDEL,FDEL,EIGREC,VRECON,SMREC,WREC,
C    1 ABFDEL)
C     GOTO 509
C**********************************************************************
C
C             CALCULATE O-LOOP EIGENVALUES
C
C**********************************************************************
C             SAVE A SINCE EIGRF DESTROYS A
165   DO 201 I = I,N
          DO 201 J = 1,N
          WM1(I,J) = A(I,J)
201   CONTINUE
          CALL EIGRF(WM1,N,10,0,Z1,Z,10,WK1,IER)
          DO 205 I = 1,N
          EIG(I) = Z1(I)
          EIGO(I) = CREAL(Z1(I))
         EIGOI(I) = DIMAG(Z1(I))
205   CONTINUE    .
         WRITE(6,2I0)
210   FORMAT(//I0X,'**** OPEN LOOP EIGENVALUES ****',//3X,'**REAL**',I5X
     *,'**IMAG**'/)
         DO 225 I = I,N
         WRITE(6,220)EIGO(I),EIGOI(I)
220   FORMAT(IX,EI2.5,10X,E12.5)
225   CONTINUE
C        RE-INITIALIZE WMI WORK MATRIX
         DO 226 I = I,N
         DO 226 J = I,N
         WM1(I,J) = 0.0
226   CONTINUE
C**********************************************************************
C
C        CHECK O-LOOP EIGENVALUES FOR CONTROLLABILITY
C
C**********************************************************************
         DO 260 I = 1,N
         X = DREAL(EIG(I))
         IF(EIGOI(I).EQ.0.D0)GOTO 227
         CALL CCNTRL(A,B,UNITYC,N,M,ICTR,EIG,I)
         GOTO 260
227   DO 22I J = I,N
         WMI(J,J) = X*UNITY(J,J)
22I   CONTINUE
         DO 230 J = I,N
         DO 230 K = I,N
         WM2(J,K) = WM1(J,K) - A(J,K)
         UTSLAM(J,K) = UNITY(J,K)
230   CONTINUE
C        CALCULATE SLAMDHA(I)
         DO 240 J = 1,N
         DO 240 K = 1,N
         SLAMDA(J,K) = WM2(J,K)
240   CONTINUE
         DO 250 J = I,N
         DO 250 K = I,M
         SLAMDA(J,K+N) = B(J,K)
250   CONTINUE
C        CALCULATE RANK OF SLAMDHA
         II = K+N
         CALL LSVDF(SLAMDA,20,N,II,UTSLAM,I0,N,SI,WK,IER)
         IF(IER.EQ.33)GOTO 251
         ICTR(I) = 1
         GOTO 260
25I   ICTR(I) = 0
260   CONTINUE
C        DISPLAY CONTROLLABILITY FLAGS FOR O-LOOP EIGENVALUES
         WRITE(6,300)
300   FORMAT(//5X,'O-LOOP EIGENVALUE        CONTROLLABILITY FLAG (1=Y)')
         DO 310 I = I,N
         WRITE(6,320) EIGO(I),EIGOI(I),ICTR(I)
320   FORMAT(IX,E10.3,IX,EI0.3,'J',I4X,I2)
310   CONTINUE
C        DECISION FOR REINSERTING EIGENSTRUCTURE
         WRITE(6,3I2)
3I2   FORMAT(/10X,'**** DESIRED EIGENSTRUCTURE HAS BEEN READ FROM ****',
     1       /I0X,'**** DATA FILE. ENTER "I" TO CHG. NOTE: CANNOT ****',
     2       /I0X,'**** CHG IF KAUTSKY ALGOR. WILL BE USED. ENTER ****',
     3       /10X,'**** "2" IF YOU DESIRE TO USE FILE DATA FOR    ****',
     4       /10X,'**** MOORE ALGOR. OR IF YOU ARE GOING TO USE   ****',
     5       /I0X,'**** KAUTSKY ALGORTIHM.                        ****',
     6 /)
         READ(*,313)IEIGEN
313   FORMAT(II)
         CALL FRTCMS('CLRSCRN ')
         IF(IEIGEN.EQ.2)GOTO 352
```

```
C**********************************************************************
C
C              INPUT DESIRED C-LOOP EIGENVALUES
C
C**********************************************************************
311   WRITE(6.325)
325   FORMAT(/5X.'**** ENTER DESIRED C-LOOP EIGENVALUES  ****'./5X.
          1            '**** REMEMBER: ONE CANNOT SHIFT THE    ****'./5X.
          2            '**** UNCCNTROLLABLE EIGENVALUE(S)..... ****'./5X.
          3            '**** ONE MUST SHIFT IN SAME OROER AS   ****'./5X.
          4            '**** WAS DISPLAYED TO MAINTAIN PROPER  ****'./5X.
          5            '**** CONTRCLLABILLITY IDENTIFICATION   ****'./5X)
            DO 330 I = 1.N
            WRITE(6.324)I
324   FORMAT(1X.'DESIRED C-LOOP EIGENVALUE ('.I1.') ='./5X.
          1          'REAL PART ..ENTER THEN IMAC PART ..ENTER'./)
            READ(*.326) X
            READ(*.326) Y
326   FORMAT(F12.5)
            EICD(I) = DCMPLX(X.Y)
330   CONTINUE
C**********************************************************************
C
C          INPUT DESIRED VALUE OF C*V
C
C**********************************************************************
335   WRITE(6.340)
340   FORMAT(/   10X.'**** INPUT DESIRED C X EICENVECTOR  ****'./10X.
          1            '**** FOR EACH EICENVALUE IN COLUMN  ****'./10X.
          2            '**** FORMAT.                        ****'./)
            DO 350 J = 1.N
            OO 350 I = 1.L
            WRITE(6.351)I.J
351   FORMAT(1X.'E('.I1.'.'.I1.') = './5X.
          1          'REAL PART ..ENTER THEN IMAC PART ..ENTER')
            READ(*.345) X
            READ(*.345) Y
345   FORMAT(F12.5)
            E(I.J) = DCMPLX(X.Y)
350   CONTINUE
C**********************************************************************
C
C               DISPLAY DESIREO EICENSTRUCTURE
C
C**********************************************************************
352   CALL FRTCMS('CLRSCRN ')
            WRITE(6.360)
360   FORMAT(//10X.'**** DESIRED EIGENSTRUCTURE FOR USE IN MOORE ****'./
          1 10X.       '**** ALGORITHMS. IGNORE IF KAUTSKY ALGORITHM ****'./
          2 10X.       '**** WILL BE SELECTED.                       ****'//
          3 .6X.'EIGENVALUES'.12X.'C * V(I)'./)
            DO 380 I = 1.N
            WRITE(6.370) EICD(I).(E(K.I).K=1.L)
370   FORMAT(1X.2(E10.3.1X).1X.8(/24X.E10.3.1X.E10.3))
380   CONTINUE
C
C**********************************************************************
C
C       CALCULATE EICENVECTCR ASSOCIATED W/EACH EIGENVALUE
C
C**********************************************************************
C          DECIDE ON MATRIX OR ALCEBRAIC SOLUTICN OR KAUTSKY ALOCORITHM
            WRITE(6.385)
385   FORMAT(///5X.'**** ENTER "1" FOR MOORE-MATRIX SOLN.   ****'./5X
          1.           '**** ENTER "2" FOR MOORE-ALCEBRAIC SOLN.****'./5X.
          2            '**** ENTER "3" FCR KAUTSKY ALGORITHM.   ****'./5X.
          3            '**** NOTE: MOORE-ALCEBRAIC SOLUTION IS  ****'./5X.
          4            '**** RECO IF L>M ANO YOU DESIRE MOORE AL****'./5X.
          5            '**** CORITHM. KAUTSKY ALCORITHM REOD    ****'./5X.
```

```
     6            '**** FOR COMPLEX EIGENVALUE ANALYSIS AND****',/5X,
     7            '**** INTERACTIVE EIGENSTRUCTURE DESIGN  ****',/5X,
     8            '**** AND FOR OUTPUT FEEDBACK PROBLEMS.  ****',/)
            READ(*,386)ISOLN
386   FORMAT(I1)
            CALL FRTCMS('CLRSCRN ')
            IF(ISOLN.EQ.1)GOTO 387
            IF(ISOLN.EQ.2)GOTO 381
            CALL KVECT(A,B,C,D,EIGD,F,N,M,L,VD,V,IFEED)
            WRITE(6,382)
382   FORMAT(/5X,'**** KAUTSKY ALGORITM COMPLETED ****',/)
            GOTO 509
381   CALL ALGSLN(A,B,C,EIGD,E,UNITY,N,M,L,V,WM3)
            WRITE(6,388)
388   FORMAT(/5X,'**** ALGEBRAIC SOLUTION COMPLETED ****',/)
            GOTO 481
387   DO 400 I = 1,N
C      REINIT SLAMDHA MATRIX
            DO 390 II= 1,10
            DO 390 JJ= 1,13
            SLAMDA(II,JJ)= 0.00
390   CONTINUE
C        REINITIALIZE WC2
            DO 401 II = 1,10
            WC2(II) = 0.00
401   CONTINUE
C        RECOMPUTE SLAMDHA MATRIX FOR EACH EIGENVALUE
            DO 395 J = 1,N
            DO 395 K = 1,N
            X = DREAL(EIGD(I))
            SLAMDA(J,K)= X*UNITY(J,K) - A(J,K)
395   CONTINUE
C        AUGMENT SLAMDA TO INCLUDE B
            DO 396 J = 1,N
            DO 396 K = 1,M
            SLAMDA(J,K+N) = B(J,K)
396   CONTINUE
C        AUGMENT SLAMDA TO INCLUDE C
            DO 410 J = 1,L
            DO 410 K = 1,N
            SLAMDA(J+N,K) = C(J,K)
410   CONTINUE
C        CALCULATE <0    E(I)> TRANSPOSE
            DO 420 J = 1,L
            X = DREAL(E(J,I))
            WC2(J+N) = X
420   CONTINUE
C        CALCULATE < V(I)  W(I) > TRANSPOSE
            I1 = N + L
C        DEBUG WRITE STATEMENT
C     WRITE(6,421)
C21   FORMAT(5X,'**** SLAMDA AUGMENTED MATRIX****',/)
C     DO 423 K1 = 1,I1
C     WRITE(6,422)(SLAMDA(K1,K2),K2=1,I1)
C22   FORMAT(1X,6(E10.2,2X))
C23   CONTINUE
C**********************************************************************
C
C               JUMP TO 680 FOR UNCONTROLLABLE EIGENVALUES
C              (PRESENTLY PROGRAM WILL NOT HANDLE THIS CASE)
C
C**********************************************************************
            IF(ICTR(I).EQ.0)GOTO 680
424   CALL LEQT1F(SLAMDA,1,I1,10,WC2,1,WKAREA,IER)
470   DO 480 J = 1,N
            V(J,I) = DCMPLX(WC2(J),0.00)
480   CONTINUE
            DO 485 J = 1,M
            WM3(J,I) = WC2(J+N)
```

```
485   CONTINUE
400   CONTINUE
C*********************************************************************
C
C        CHECK FOR INDEPENDENCE OF EIGENVECTORS COMPUTED VIA
C        MOORE ALGORITHMS
C
C*********************************************************************
C        CALCULATE RANK OF V
C   LSVDF REPLACES 'V' WITH THE ORTHOGONAL 'V' MATRIX A LA SINGULAR
C   VALUE DECOMPOSITION. THEREFORE 'SAVE' REAL PART.
481   DO 482 I = 1,N
          DO 482 J = 1,N
          VR(I,J) = DREAL(V(I,J))
          VRS(I,J) = VR(I,J)
          VRT(I,J) = UNITY(I,J)
482   CONTINUE
          CALL LSVDF(VR,10,N,N,VRT,10,N,S2,WK1,IER)
          IF(IER.EQ.33)GOTO 490
          GOTO 501
490   WRITE(6,500)
500   FORMAT(//5X,'****EIGENVECTORS ARE NOT INDEPENDENT, MUST ****',/5X,
     1            '**** CHANGE C X EIGENVECTOR (I) OR CHG IN- ****',/)
          IF(ISOLN.EQ.3)GOTO 680
          GOTO 335
C*********************************************************************
C
C        CALCULATE FEEDBACK GAINS
C
C*********************************************************************
501   CALL LINV2F(VRS,N,10,WM4,1,WAREA1,IER)
          WRITE(6,507)
507   FORMAT(/5X,'**** EIGENVECTOR MAT HAS BEEN INVERTED, F GAINS NEXT *
     1***',/)
          CALL VMULFF(WM3,WM4,M,N,N,10,10,F,10,IER)
          WRITE(6,508)
508   FORMAT(/5X,'**** F GAINS COMPUTED ****',/)
C        ASSIGN PROPER SENSE TO FEEDBACK GAINS
          DO 506 I = 1,M
          DO 506 J = 1,N
          F(I,J) = - F(I,J)
506   CONTINUE
C*********************************************************************
C
C    CALCULATE A + BF OR A + BFC OR A + B*INV(I - FD)*FC
C
C*********************************************************************
509   IF(IFEED.EQ.1)GOTO 513
          IF(IFEED.EQ.2)GOTO 514
            CALL VMULFF(F,C,M,L,N,10,10,WM7,10,IER)
            CALL VMULFF(F,D,M,L,M,10,10,WM8,10,IER)
            DO 516 I = 1,M
            DO 516 J = 1,M
516   WM9(I,J) = UNITY(I,J) - WM8(I,J)
            CALL LINV2F(WM9,M,10,WM10,0,WAREA2,IER)
            IF(IER.NE.129)GOTO 518
            WRITE(6,519)
519   FORMAT(/5X,'**** I - FD IS SINGULAR. RESULTS ARE INACCURATE ****'
     1 /)
518   CALL VMULFF(WM10,WM7,M,M,N,10,10,WM11,10,IER)
          CALL VMULFF(B,WM11,N,M,N,10,10,WM12,10,IER)
          DO 517 I = 1,N
          DO 517 J = 1,N
517   ABF(I,J) = A(I,J) + WM12(I,J)
          GOTO 512
514   CALL VMULFF(B,F,N,M,L,10,10,WM5,10,IER)
          CALL VMULFF(WM5,C,N,L,N,10,10,WM6,10,IER)
          DO 515 I = 1,N
          DO 515 J = 1,N
```

```
515   ABF(I,J) = A(I,J) + WM6(I,J)
            GOTO 512
513   CALL VMULFF(B,F,N,M,N,10,10,WM5,10,IER)
            DO 510 I = 1,N
            DO 510 J = 1,N
            ABF(I,J) = A(I,J) + WM5(I,J)
510   CONTINUE
C***********************************************************************
C
C     CHECK FOR CONSERVATION OF C*V DURING DESIGN ITERATIONS
C
C***********************************************************************
            IF(ISOLN.EQ.3)GOTO 512
511   CALL VMULFF(C,VRS,L,N,N,10,10,CV,10,IER)
512   WRITE(6,520)
520   FORMAT(/,'**** IF SINGULAR VALUE ANALYSIS DESIRED, ENTER "1" ****'
          1,/,      '****           OTHERWISE ENTER "2"                  ****'
          2)
            READ(*,530)ISVA
530   FORMAT(I1)
            CALL FRTCMS('CLRSCRN ')
            IF(ISVA.EQ.2)GOTO 540
C***********************************************************************
C
C              SINGULAR VALUE ANALYSIS OF DESIGN
C
C***********************************************************************
C31   F(1,1) = -14.18705
C     F(1,2) = 3.5569
C     F(1,3) = 77.9429
C     F(1,4) = -61.009
C     F(2,1) = -0.05329
C     F(2,2) = -2.61465
C     F(2,3) = .20053
C     F(2,4) = -15.58603
531   CALL SVA(A,B,C,D,F,UNITY,N,M,L,W,SVM,SVMAX,SVM1,SVMAX1,IFEED)
C***********************************************************************
C
C                  DISPLAY RESULTS
C
C***********************************************************************
540   CALL FRTCMS('CLRSCRN ')
            WRITE(6,600)
600   FORMAT(///15X,'********** DESIGN RESULTS **********',//)
            IF(ISOLN.EQ.3)GOTO 633
632   WRITE(6,601)
601   FORMAT(//5X,'**** W MATRIX COMPUTED DURING MOORE SOLN ****'/)
            DO 602 I = 1,M
            WRITE(6,603)(WM3(I,J),J=1,N)
603   FORMAT(1X,6(E12.5,1X))
602   CONTINUE
633   WRITE(6,631)
631   FORMAT(//5X,'**** C-LOOP MATRIX, A + BF ****',/)
            DO 640 I = 1,N
            WRITE(6,603)(ABF(I,J),J=1,N)
640   CONTINUE
            WRITE(6,650)
650   FORMAT(//5X,'**** FEEDBACK MATRIX ****',//)
            DO 670 I = 1,M
            WRITE(6,660)(F(I,J),J=1,L)
660   FORMAT(1X,5(2X,E12.5))
670   CONTINUE
C        SAVE 'F' SINCE LSVDF DESTROYS F
            DO 651 I = 1,M
            DO 651 J = 1,L
651   FSAVE(I,J) = F(I,J)
            CALL LSVDF(FSAVE,10,M,L,UIF,10,0,SF,WK,IER)
            X = SF(1)
            WRITE(6,695)X
```

```
695    FORMAT(/5X,'SPECTRAL NORM = ',E12.5)
             CALL EIGRF(ABF,N,10,1,Z5,Z4,10,WK4,IER)
C         FINO CONDITION NUMBER OF OESIGN EIGENVECTOR MATRIX
             CALL COND(Z4,N,X)
             WRITE(6,692)
692    FORMAT(/5X,'**** DESIGN EIGENVECTOR MATRIX ****',/)
             DO 693 I = 1,N
             WRITE(6,620)(Z4(I,J),J=I,N)
693    CONTINUE
             WRITE(6,694)X
694    FORMAT(/5X,'2 NORM CONDITION NUMBER = ',E12.5)
C          NORMALIZE EIGENVECTOR MATRIX
             DO 671 J = 1,N
             X = 0.D0
             DO 672 I = 1,N
672    X = X + DREAL(Z4(I,J))**2 + OIMAG(Z4(I,J))**2
             DO 671 I = 1,N
671    Z4(I,J) = Z4(I,J)/DSQRT(X)
             WRITE(6,610)
610    FORMAT(/5X,'**** NORMALIZED DESIGN EIGENVECTOR MATRIX ****',/)
             DO 630 I = 1,N
             WRITE(6,620)(Z4(I,J),J=I,N)
620    FORMAT(8(1X,E9.2))
630    CONTINUE
             CALL COND(Z4,N,XN)
             WRITE(6,694)XN
             WRITE(6,690)
690    FORMAT(/5X,'**** DESIGN C-LOOP EIGENVALUES ****',/I3X,
          1 'REAL PART',6X,'IMAG PART',/)
             DO 691 I = 1,N
             WRITE(6,156)I,Z5(I)
691    CONTINUE
             IF(ISOLN.EQ.3)GOTO 915
912    WRITE(6,900)
900    FORMAT(//5X,'**** E=<C><V> MATRIX AFTER DESIGN COMPLETE ****',/
          15X,          '**** NOTE: PGM CHECK FOR MOORE ALGORITHM   ****'/)
             DO 910 I = 1,L
             WRITE(6,920)(CV(I,J),J=I,N)
920    FORMAT(1X,6(2X,E12.5))
910    CONTINUE
915    IF(ISVA.EQ.2)GOTO 680
             WRITE(6,930)
930    FORMAT(//15X,'**** SINGULAR VALUE ANALYSIS OF OESIGN *****',//1X,
          1                '    FREQ              I + FG                        I + GF'
          2,//,2X,       ' RAD/SEC   MAX SIG       MIN SIG      MAX SIG       MIN
          3 SIG',/)
             DO 940 I = 1,210,10
             WRITE(6,950) W(I),SVMAX(I),SVM(I),SVMAXI(I),SVMI(I)
950    FORMAT(3X,F7.2,1X,4(E12.5,1X))
940    CONTINUE
C           GENERATE OATA FILE FOR S.V. PLOTTING
             CALL SVPLOT(SVM,SVM1,W)
             WRITE(6,1000)
1000   FORMAT(//5X,'**** IF YOU ARE AT A TEK618 CONSOLE ANO DESIRE ****',
          1         /5X,'**** A PLOT OF MINIMUM S.V. OF THE RETURN DIFF ****',
          2         /5X,'**** MATRIX JUST COMPUTEO; TYPE THE FOLLOWING  ****',
          3         /5X,'**** AFTER EIGENS COMPLETION.................. ****',
          4         /5X,'**** "PLOT" FOLLOWEO BY "OISSPLA SVPLOT FORTRAN"****'
          5        /)
680    IF(IRECON.EQ.2)GOTO 960
             WRITE(6,970)
970    FORMAT(/5X,'**** OAMAGEO B MATRIX ****',/)
             DO 971 I = 1,N
971    WRITE(6,90)(BDEL(I,J),J=1,M)
             WRITE(6,972)
972    FORMAT(/5X,'**** RECONFIGURED FEEDBACK GAINS ****',/)
             OO 973 I = 1,M
973    WRITE(6,90)(FDEL(I,J),J=1,L)
             WRITE(6,975)
```

```
975   FORMAT(/5X,'**** RECONFIGURED NORMALIZED EIGENVECTORS ****',/)
            WRITE(6,620)((VRECON(I,J),J=1,N),I=1,N)
            WRITE(6,976)
976   FORMAT(/5X,'**** RECONFIGURED EIGENVALUES ****',/)
            WRITE(6,156)(I,EIGREC(I),I=1,N)
            WRITE(6,977)SMREC,WREC
977   FORMAT(/5X,'**** MIN S.V = ',E12.5,' ****',/5X,'**** AT FREQ = ',
         1E12.5,' ****',/)
            WRITE(6,978)
978   FORMAT(/5X,'**** RECONFIGURED C-LCOP MATRIX ****',/)
            DO 979 I = 1,N
979   WRITE(6,603)(ABFDEL(I,J),J=1,N)
960   CALL RESULT(A,B,C,F,FDEL,BDEL,Z4,Z5,L,M,N)
            END
C*********************************************************************** .
C
C             SUBROUTINE SVA PERFORMS SINGULAR VAULE ANALYSIS OF
C             THE RETURN DIFFERENCE MATRIX OF THE DESIGNED SYSTEM.
C             IMSL ROUTINES RECD: LEQT1C
C             EIGENS SUBROUTINES REQD: CMATML,CSVD
C             CDR V.F. GAVITO VER 1.0  JANUARY, 1986
C
C***********************************************************************
            SUBROUTINE SVA(A,B,C,D,F,UNITY,N1,M1,LI,W,SVM,SVMAX,SVM1,SVMAX1,
          1 IFEED)
            IMPLICIT REAL*8(A-H,P-Z)
            REAL*B A(10,10),B(10,10),C(10,10),UNITY(10,10),W(2000),SVM(2000),
          I       S1(10),X1,YI,F(I0,10),SVMAX(2000),X,S2(10),SVM1(2000),
          2       SVMAX1(2000),WA(10),D(10,10)
            COMPLEX*16AC(10,10),BC(10,I0),CC(I0,10),WMC2(10,10),WMC3(10,10),
          1            GC(10,10),UI(10,10),VI(10,I0),RDM(10,10),WMC4(10,10),
          2            WMC5(I0,10),FC(10,10),RDM1(10,10),UI1(10,10),VI1(I0,10)
          3,           DC(I0,10)
            COMPLEX*16 WMC1(I0,10),UNITYC(I0,I0)
            INTEGER N1,MI,L1,IFEED
            DO 10 I = I,10
            WA(I) = 0.D0
            S1(I) = 0.D0
            S2(I) = 0.D0
            DO I0 J = I,I0
            AC(I,J) = (0.D0,0.D0)
            RDM(I,J) = (0.D0,0.D0)
            RDM1(I,J) = (0.D0,0.D0)
            CC(I,J) = (0.D0,0.D0)
            UI(I,J) = (0.D0,0.D0)
            VI(I,J) = (0.D0,0.D0)
            UI1(I,J) = (0.D0,0.D0)
            VI1(I,J) = (0.D0,0.D0)
            UNITYC(I,J) = (0.D0,0.D0)
            WMCI(I,J) = (0.D0,0.D0)
            WMC2(I,J) = (0.D0,0.D0)
            WMC3(I,J) = (0.D0,0.D0)
            WMC4(I,J) = (0.D0,0.D0)
            WMC5(I,J) = (0.D0,0.D0)
            BC(I,J) = (0.D0,0.D0)
            DC(I,J) = (0.D0,0.D0)
            GC(1,J) = (0.D0,0.D0)
            FC(1,J) = (0.D0,0.D0)
10    CONTINUE
            DO 45 I = 1,2000
            SVM(I) = 0.D0
            SVMAX(I) = 0.D0
            SVM1(I) = 0.D0
            SVMAX(I) = 0.D0
45    CONTINUE
C        FILL UP COMPLEX MATRICS WITH PROPER DATA
            DO 100 I = 1,N1
            UNITYC(I,I) = (1.D0,0.D0)
            WMC5(I,I) = (1.D0,0.D0)
         DO 100 J = 1,N1
         X = A(I,J)
         AC(I,J) = DCMPLX(X,0.D0)
100   CONTINUE
         DO 200 I = 1,N1
         DO 200 J = 1,M1
         X = B(I,J)
         Y = F(J,I)
         BC(I,J) = DCMPLX(X,0.D0)
         FC(J,I) = DCMPLX(Y,0.D0)
200   CONTINUE
         DO 300 I = 1,L1
         DO 300 J = 1,N1
         X = C(I,J)
         CC(I,J) = DCMPLX(X,0.D0)
300   CONTINUE
         DO 210 I = 1,L1
         DO 210 J = 1,M1
210   DC(I,J) = DCMPLX(D(I,J),0.D0)
C        AUGMENT (IF REQD) C TO INVOKE A SQUARE RETURN DIFF MATRIX
C        PRESENTLY THE PROGRAM AUGMENTS WITH THE IDENTITY
C     IF(L1.EQ.N1)GOTO 350
C     INDEX = N1 - L1
C     DO 325 J = 1,INDEX
C     CC(J+L1,J+L1) = (1.D0,0.D0)
C25   CONTINUE
350   W(1) = 1.D-01
C      DISPLAY AUGMENTED F MATRIX FOR SVD
C     WRITE(6,310)
C10   FORMAT(15X,'**** AUGMENTED F MATRIX FOR SVD COMP ****',/)
C     DO 311 J = 1,M1
C     DO 311 K = 1,N1
C     X2 = DREAL(FC(J,K))
C     X3= DIMAG(FC(J,K))
C     WRITE(6,312)J,K,X2,X3
C12   FORMAT(2X,'FC(',I2,',',I2,') = ',E12.5,' + J',E12.5)
C11   CONTINUE
         DO 500 I = 1,210
         DO 400 J = 1,N1
         DO 400 K = 1,N1
         X = W(I)
         WMC1(J,K) = DCMPLX(0.D0,X) * UNITYC(J,K) - AC(J,K)
400   CONTINUE
C       **********IMPORTANT NOTE************
C       *   LEQT1C DESTROYS WMC1 AND      *
C       * REPLACES UNITYC WITH THE INVERSE*
C       *  OF WMC1                        *
C       ***********************************
         CALL LEQT1C(WMC1,N1,10,UNITYC,N1,10,0,WA,IER)
C     WRITE(6,401)
C01   FORMAT(1X,' <SI - A>-I ',/)
C     DO 402 J = 1,N1
C     WRITE(6,4100)(UNITYC(J,K),K=1,N1)
C02   CONTINUE
         CALL CMATML(UNITYC,BC,N1,N1,M1,WMC3)
         CALL CMATML(CC,WMC3,L1,N1,M1,GC)
C         COMPUTE <F> * <G>
C     DO 8000 J = 1,M1
C     WRITE(6,4100)(FC(J,K),K=1,N1)
C000  CONTINUE
         IF(IFEED.EQ.1)GOTO 410
         IF(IFEED.EQ.2)GOTO 410
         DO 412 J = 1,L1
         DO 412 K = 1,M1
412   GC(J,K) = DC(J,K) + GC(J,K)
410   CALL CMATML(FC,GC,M1,L1,M1,WMC2)
C     WRITE(6,4000)
C000  FORMAT(1X,'<SI - A> **-1  * B',/)
C     DO 4001 J = 1,N1
```

```
C     WRITE(6.4100)(WMC3(J.K).K=1.M1)
C100  FORMAT(8(E9.2.1X.E9.2))
C001  CONTINUE
C     WRITE(6.5000)
C000  FORMAT(1X.'C * INV<SI - A> * 8'./)
C     DO 5001 J = 1.N1
C     WRITE(6.4100)(GC(J.K).K=1.M1)
C001  CONTINUE
C     WRITE(6.6000)
C000  FORMAT(1X.' F * G './)
C     DO 6001 J = 1.M1
C     WRITE(6.4100)(WMC2(J.K).K=1.M1)
C001  CONTINUE
C          CCMPUTE <G> * <F>
            CALL CMATML(GC.FC.L1.M1.L1.WMC4)
C        RESTORE THE COMPLEX IDENTITY MATRIX
            DO 450 J = 1.N1
            DO 450 K = 1.N1
            X = UNITY(J.K)
            UNITYC(J.K) = DCMPLX(X.0.D0)
450   CONTINUE
C        COMPUTE <I> + <F> * <G>
            DO 600 J = 1.M1
            DO 600 K = 1.M1
            RDM(J.K) = UNITYC(J.K) + WMC2(J.K)
600   CONTINUE
C        COMPUTE <I> + <G> * <F>
            DO 650 J = 1.L1
            DO 650 K = 1.L1
            RDM1(J.K) = UNITYC(J.K) + WMC4(J.K)
650   CONTINUE
C     WRITE(6.610)I.W(I)
C10   FORMAT(2X.'W('.I2.') = '.E12.5)
C     WRITE(6.611)
C11   FORMAT(2X.'**** RETURN DIFFERENCE MATRIX ****'./)
C     DO 612 J = 1.M1
C     DO 612 K = 1.M1
C     X1=DREAL(RDM(J.K))
C     Y1=DIMAG(RDM(J.K))
C     WRITE(6.613)J.K.X1.Y1
C13   FORMAT(1X.'RDM ('.I2.'.'.I2.') = '.E12.5.' + J'.E12.5)
C12   CONTINUE
C        COMPUTE CSVD OF <I> + <F> * <G>
            CALL CSVD(RDM.10.10.M1.M1.0.M1.M1.S1.UI.VI)
C        COMPUTE CSVD OF <I> + <G> * <F>
            CALL CSVD(RDM1.10.10.L1.L1.0.L1.L1.S2.UI1.VI1)
C     WRITE(6.800)I
C00   FORMAT(/5X.'SINGULAR VALUES FOR W('.I2.') ='./)
C     WRITE(6.700)(S1(J).J=1.L1)
C00   FORMAT(1X.6(E12.5.1X))
C     WRITE (6.615) S1(L1)
C15   FORMAT(/2X.'MIN SINGULAR VALUE = '.E12.5)
            SVMAX(I) = S1(1)
            SVMAX1(I) = S2(1)
            IF(S1(M1).LE.0.)GOTO 617
            IF(S2(L1).LE.0.D0)GOTO 617
            SVM(I) = S1(M1)
            SVM1(I) = S2(L1)
            GOTO 619
617   WRITE(6.618)
618   FORMAT(/12X.'******* RTN DIFF MATRIX IS RANK DEFICIENT*******')
            SVM(I) = 9.999999D+09
619   W(I+1) = W(I) +.50
500   CONTINUE
            RETURN
            END
C**********************************************************************
C
C              SUBROUTINE CMATML (COMPLEX MATRIX MULTIPLICATION)
```

```
C
C        COMPUTES: YY = AA * BB
C        IA = # OF ROWS IN AA
C        LL = # OF ROWS IN SB AND # OF COLUMNS IN AA
C        IB = # OF COLUMNS IN BB
C*****************************************************************************
           SUBROUTINE CMATML(AA,SB,1A,LL,IB,YY)
           CCMPLEX*16 AA(10,I0),BB(10,I0),YY(I0,10)
           INTEGER IA,LL,IB
           DO 30 I = 1,IA
           DO 20 J = 1,IB
           YY(1,J) = (0.D0,0.D0)
           DO 10 INDEX = 1,LL
           YY(1,J) = YY(I,J) + AA(I,INDEX) * BB(INDEX,J)
I0     CONTINUE
20     CONTINUE
30     CCNTINUE
           RETURN
           END
C*****************************************************************************
C
C              SUBROUTINE CSVD
C       COMPLEX SINGULAR VALUE DECOMPOSITION SUBROUTINE
C
C*****************************************************************************
           SUBROUTINE CSVD (A,MMAX,NMAX,M,N,IP,NU,NV,S,U,V)
           IMPLICIT REAL*8(A-H,P-Z)
           CCMPLEX*16A(MMAX,1),U(MMAX,1),V(NMAX,1)
           INTEGER M,N,IP,NU,NV
           REAL*8 S(1)
           COMPLEX*16 Q,R
           REAL*8 B(100),C(100),T(100)
           DATA ETA,TOL/1.5D-8,1.D-31/
           NP=N+1P
           NI=N+1
C
C  HOUSEHOLDER REDUCTION
           C(1)=0.D0
           K=1
10     K1=K+1
C
C   ELIMINATION OF A(I,K),I=K+1,.....M
           Z=0.D0
           DO 20 I=K,M
20     Z=Z+DREAL(A(I,K))**2+DIMAG(A(I,K))**2
           B(K)=0.D0
           IF (Z.LE.TOL) GO TO 70
           Z=DSQRT(Z)
           B(K)=Z
           W=CDABS(A(K,K))
           Q=(I.D0,0.D0)
           IF (W.NE.0.D0) Q=A(K,K)/W
           A(K,K)=Q*(Z+W)
           IF (K.EQ.NP) GO TO 70
           DO 50 J=K1,NP
           Q=(0.D0,0.D0)
           DO 30 I=K,M
30     C=Q+DCONJG(A(I,K))*A(1,J)
           Q=Q/(Z*(Z+W))
           DO 40 I=K,M
40     A(I,J)=A(I,J)-Q*A(1,K)
50     CONTINUE
C
C   PHASE TRANSFORMATICN
           Q=-DCONJG(A(K,K))/CDABS(A(K,K))
           DO 60 J=K1,NP
60     A(K,J)=Q*A(K,J)
C
C   ELIMINATION OF A(K,J),J=K+2,....N
```

```
70    IF (K.EQ.N) GO TO 140
            Z=0.D0
            DO 80 J=K1,N
80    Z=Z+DREAL(A(K,J))**2+DIMAG(A(K,J))**2
            C(K1)=0.D0
            IF (Z.LE.TOL) GO TO 130
            Z=DSQRT(Z)
            C(K1)=Z
            W=CDABS(A(K,K1))
            Q=(1.D0,0.D0)
            IF (W.NE.0.D0) Q=A(K,K1)/W
            A(K,K1)=Q*(Z+W)
            DO 110 I=K1,M
            Q=(0.D0,0.D0)
            DO 90 J=K1,N
90    Q=Q+DCONJG(A(K,J))*A(I,J)
            Q=Q/(Z*(Z+W))
            DO 100 J=K1,N
100   A(I,J)=A(I,J)-Q*A(K,J)
110   CONTINUE
C
C   PHASE TRANSFORMATION
            Q=-DCONJG(A(K,K1))/CDABS(A(K,K1))
            DO 120 I=K1,M
120   A(I,K1)=A(I,K1)*Q
130   K=K1
            GO TO 10
C
C  TOLERENCE FOR NEGLIGIBLE ELEMENTS
140   EPS=0.D0
            DO 150 K=1,N
            S(K)=B(K)
            T(K)=C(K)
150   EPS=DMAX1(EPS,S(K)+T(K))
            EPS=EPS*ETA
C
C   INITIALIZATION OF U AND V
            IF (NU.EQ.0) GO TO 180
            DO 170 J=1,NU
            DO 160 I=1,M
160   U(I,J)=(0.D0,0.D0)
170   U(J,J)=(1.D0,0.D0)
180   IF (NV.EQ.0) GO TO 210
            DO 200 J=1,NV
            DO 190 I=1,N
190   V(I,J)=(0.D0,0.D0)
200   V(J,J)=(1.D0,0.D0)
C
C  QR DIAGONALIZATION
210   DO 380 KK=1,N
            K=N1-KK
C
C   TEST FOR SPLIT
220   DO 230 LL=1,K
            L=K+1-LL
            IF (DABS(T(L)).LE.EPS) GO TO 290
            IF (DABS(S(L-1)).LE.EPS) GO TO 240
230   CONTINUE
C
C   CANCELLATION OF E(L)
240   CS=0.D0
            SN=1.D0
            L1=L-1
            DO 280 I=L,K
            F=SN*T(I)
            T(I)=CS*T(I)
            IF (DABS(F).LE.EPS) GO TO 290
            H=S(I)
            W=DSQRT(F*F+H*H)
            S(I)=W
            CS=H/W
            SN=-F/W
            IF (NU.EQ.0) GO TO 260
            DO 250 J=1,N
            X=DREAL(U(J,L1))
            Y=DREAL(U(J,I))
            U(J,L1)=DCMPLX(X*CS+Y*SN,0.D0)
250   U(J,I)=DCMPLX(Y*CS-X*SN,0.D0)
260   IF (NP.EQ.N) GO TO 280
            DO 270 J=N1,NP
            Q=A(L1,J)
            R=A(I,J)
            A(L1,J)=Q*CS+R*SN
270   A(I,J)=R*CS-Q*SN
280   CONTINUE
C
C   TEST FOR CONVERGENCE
290   W=S(K)
            IF (L.EQ.K) GO TO 360
C
C   ORIGIN SHIFT
            X=S(L)
            Y=S(K-1)
            G=T(K-1)
            H=T(K)
            F=((Y-W)*(Y+W)+(G-H)*(G+H))/(2.D0*H*Y)
            G=DSQRT(F*F+1.D0)
            IF (F.LT.0.D0) G=-G
            F=((X-W)*(X+W)+(Y/(F+G)-H)*H)/X
C
C   QR STEP
            CS=1.D0
            SN=1.D0
            L1=L+1
            DO 350 I=L1,K
            G=T(I)
            Y=S(I)
            H=SN*G
            G=CS*G
            W=DSQRT(H*H+F*F)
            T(I-1)=W
            CS=F/W
            SN=H/W
            F=X*CS+G*SN
            G=G*CS-X*SN
            H=Y*SN
            Y=Y*CS
            IF (NV.EQ.0) GO TO 310
            DO 300 J=1,N
            X=DREAL(V(J,I-1))
            W=DREAL(V(J,I))
            V(J,I-1)=DCMPLX(X*CS+W*SN,0.D0)
300   V(J,I)=DCMPLX(W*CS-X*SN,0.D0)
310   W=DSQRT(H*H+F*F)
            S(I-1)=W
            CS=F/W
            SN=H/W
            F=CS*G+SN*Y
            X=CS*Y-SN*G
            IF (NU.EQ.0) GO TO 330
            DO 320 J=1,N
            Y=DREAL(U(J,I-1))
            W=DREAL(U(J,I))
            U(J,I-1)=DCMPLX(Y*CS+W*SN,0.D0)
320   U(J,I)=DCMPLX(W*CS-Y*SN,0.D0)
330   IF (N.EQ.NP) GO TO 350
            DO 340 J=N1,NP
            Q=A(I-1,J)
            R=A(I,J)
            A(I-1,J)=O*CS+R*SN
340   A(I,J)=R*CS-O*SN
350   CONTINUE
            T(L)=0.D0
            T(K)=F
            S(K)=X
            GO TO 220
C
C  CONVERGENCE
360   IF (W.GE.0.D0) GO TO 380
            S(K)=-W
            IF (NV.EO.0) GO TO 380
            DO 370 J=1,N
370   V(J,K)=-V(J,K)
380   CONTINUE
C
C  SORT SINGULAR VALUES
            DO 450 K=1,N
            G=-1.D0
            J=K
            DO 390 I=K,N
            IF (S(I).LE.G) GO TO 390
            G=S(I)
            J=I
390   CONTINUE
            IF (J.EQ.K) GO TO 450
            S(J)=S(K)
            S(K)=G
            IF (NV.EO.0) GO TO 410
            DO 400 I=1,N
            O=V(I,J)
            V(I,J)=V(I,K)
400   V(I,K)=O
410   IF (NU.EQ.0) GO TO 430
            DO 420 I=1,N
            Q=U(I,J)
            U(I,J)=U(I,K)
420   U(I,K)=O
430   IF (N.EO.NP) GO TO 450
            DO 440 I=N1,NP
            O=A(J,I)
            A(J,I)=A(K,I)
440   A(K,I)=O
450   CONTINUE
C
C  BACK TRANSFORMATION
            IF (NU.EQ.0) GO TO 510
            DO 500 KK=1,N
            K=N1-KK
            IF (B(K).EO.0.D0) GO TO 500
            O=-A(K,K)/CDABS(A(K,K))
            DO 460 J=1,NU
460   U(K,J)=O*U(K,J)
            DO 490 J=1,NU
            O=(0.D0,0.D0)
            DO 470 I=K,M
470   O=O+DCONJG(A(I,K))*U(I,J)
            O=O/(CDABS(A(K,K))*B(K))
            DO 480 I=K,M
480   U(I,J)=U(I,J)-O*A(I,K)
490   CONTINUE
500   CONTINUE
510   IF (NV.EO.0) GO TO 570
            IF (N.LT.2) GO TO 570
            DO 560 KK=2,N
            K=N1-KK
            K1=K+1
            IF (C(K1).EO.0.D0) GO TO 560
```

```
          Q=-DCONJG(A(K,K1))/CDABS(A(K,K1))
          DO 520 J=1,NV
520   V(K1,J)=Q*V(K1,J)
          DO 550 J=1,NV
          Q=(0.D0,0.D0)
          DO 530 I=K1,N
530   Q=Q+A(K,I)*V(I,J)
          Q=Q/(CDABS(A(K,K1))*C(K1))
          DO 540 I=K1,N
540   V(I,J)=V(I,J)-C*DCONJG(A(K,I))
550   CONTINUE
560   CONTINUE
570   RETURN
          END
C********************************************************************
C
C      SUBROUTINE SVPLDT: GENERATES DATA FILE FOR USING DISSPLA
C                 TO PLOT MINIMUM SINGULAR VALUE OF THE RETURN
C                 DIFFERENCE MATRIX VS. FREQUENCY
C
C********************************************************************
          SUBRDUTINE SVPLOT(SVM,SVM1,W)
          REAL*8 W(2000),SVM(2000),SVM1(2000)
          DIMENSION WP(2000),SVMP(2000),SVM1P(2000)
C      CONVERT DATA TO SINGLE PRECISION FOR DISSPLA COMPATIBILITY
          DO 10 I = 1,2000
          WP(I) = SNGL(W(I))
          SVMP(I) = SNGL(SVM(I))
          SVM1P(I) = SNGL(SVM1(I))
10    CONTINUE
          DO 20 I = 1,2000,2
          WRITE(7,30)WP(I),SVMP(I),SVM1P(I),WP(I+1),SVMP(I+1),SVM1P(I+1)
30    FORMAT(3(E12.5,1X,E12.5))
20    CONTINUE
          RETURN
          END
C********************************************************************
C                                                                   *
C           SUBRDUTIN KVECT: CALCULATES FEEDBACK GAINS VIA          *
C                       KAUTSKY, ET. AL. ALGORITHM (1985)           *
C                                                                   *
C********************************************************************
          SUBRCUTINE KVECT(A,B,C,D,EIGD,F,N1,M1,L1,VD,V,IFEED)
          IMPLICIT REAL*8(A-H,P-Z)
          REAL*8 A(10,10),B(10,10),C(10,10),F(10,10),UT(10,10),CS(10,10),
        1 S(10),WK(20),BS(10,10),SM(10,10),VT(10,10),CUT(10,10),SC(10),
        2 RZ(10,10),WAPEA1(250),WAREA2(250),D(10,10),CSS(10,10),
        3 UTV(10,10),RZINV(10,10),V1(10,10),WK4(20),CSSINV(10,10),
        4 S1(10),S2(10),WK1(20),WK2(20),WA(20),S8(10),WK3(50),BG1(10,10),
        5 VDST(10,10),VD1ST(10,10),U(10,10),U0(10,10),U1T(10,10),U1(10,10),
        6 UNITY(10,10),W(2000),SVM(2000),SVMAX(2000),SVM1(2000),
        7 U0T(10,10),RNULL(10,10),SVMAX1(2000),BPI(10,10),BSI(10,10),
        8 SMC(10,10),SV(10,10),BCHK(10,10),WAREA3(250),SX(10),CG1(10,10)
          CCMPLEX*16 EIGD(10),V(10,10),UNITYC(10,10),AML(10,10),U1TC(10,10)
        1 ,RNULLC(10,10),WV(10),WV1C(10),RES(10,10),VSAVE(10,10),
        2 VDS(10,10),VD1S(10,10),UVDS(10,10),VVDS(10,10),UVD1S(10,10),
        3 VVD1S(10,10),Z1,VD(10,10),EIGDM(10,10),EVI(10,10),VEV1(10,10),
        4 VMA(10,10),U0TC(10,10),UTVC(10,10),RZ1NVC(10,10),FC(10,10),
        5 VD1(10,10),VIV(10,10),U0C(10,10),C0TC(10,10),C1TC(10,10),
        6 RZU(10,10),BPIC(10,10),Z1NVCC(10,10),FCZ(10,10),DCFC(10,10),
        7 FCS(10,10),DC(10,10),RZC(10,10),C1NVZ(10,10),DCFC1(10,10)
        8,XVS(10,10),XV(10,10),XVU(10,10),XVV(10,10),SXC(10,10),
        9 X0T(10,10),SXX(10,10),RU(10,10),RV(10,10),RVV(10,10),RX(10,10)
          CCMPLEX*16 RXZ(10,10),XVT(10,10),XE(10,10),AC(10,10),AX(10,10),
        1 XCHECK(10,10),RVU(10,10)
          INTEGER N1,M1,L1,IWK(1000),IC(10),IDG(1),IDG1(20),IC1(20),
        1 IROW(10,10),IFLT(10),IELEM(10),ILCG,IFEED
          REAL*4 SRCBJ,EPS,SWV(10),VLB(100),VUB(100),RA(20,40),G(1),DF(20),
        1 WK3(3000),XBJ(100),CDBJ,GC(20),E,UR,UC,EPSR,EPSC,E1,E2
```

```
      CHARACTER*80 TITLE
C**********************************************************************
C
C        INITIALIZE MATRICES & VARIABLES
C
C**********************************************************************
      EPS = 0.
      E = 0.
      E1 = 0.
      E2 = 0.
      DO 10 I = 1,10
      S(I) = 0.D0
      S1(I) = 0.D0
      S2(I) = 0.D0
      SB(I) = 0.D0
      SC(I) = 0.D0
      SX(I) = 0.D0
      WV(I) = 0.D0
      DF(I) = 0.
      IFLT(I) = 0
      SWV(I) = 0.
      DO 10 J = 1,10
      UT(I,J) = 0.D0
      V1(I,J) = 0.D0
      CUT(I,J) = 0.D0
      CS(I,J) = 0.D0
      SM(I,J) = 0.D0
      AC(I,J) = (0.D0,0.D0)
      BPI(I,J) = 0.D0
      BSI(I,J) = 0.D0
      VDI(I,J) = (0.D0,0.D0)
      EIGDM(I,J) = (0.D0,0.D0)
      EVI(I,J) = (0.D0,0.D0)
      VEVI(I,J) = (0.D0,0.D0)
      VMA(I,J) = (0.D0,0.D0)
      UTV(I,J) = 0.D0
      UTVC(I,J) = (0.D0,0.D0)
      VDS(I,J) = 0.D0
      VDST(I,J) = 0.D0
      VDIS(I,J) = 0.D0
      VDIST(I,J) = 0.D0
      U(I,J)= 0.D0
      U0(I,J) = 0.D0
      U0T(I,J) = 0.D0
      U0TC(I,J) = (0.D0,0.D0)
      U1(I,J) = 0.D0
      U1T(I,J) = 0.D0
      U1TC(I,J) = (0.D0,0.D0)
      RNULL(I,J) = 0.D0
      RES(I,J) = (0.D0,0.D0)
      RZINV(I,J) = 0.D0
      RZINVC(I,J) = (0.D0,0.D0)
      UNITY(I,J) = 0.D0
      AML(I,J) = (0.D0,0.D0)
      SMC(I,J) = 0.D0
      SV(I,J) = 0.D0
      BCHK(I,J) = 0.D0
      RA(I,J) = 0.
      IROW(I,J) = 0
      UNITYC(I,J) = (0.D0,0.D0)
      C0TC(I,J) = (0.D0,0.D0)
      ZINVCC(I,J) = (0.D0,0.D0)
      CSS(I,J) = 0.D0
      CSSINV(I,J) = 0.D0
      CINVZ(I,J) = (0.D0,0.D0)
      BPIC(I,J) = (0.D0,0.D0)
      RZU(I,J) = (0.D0,0.D0)
      FCZ(I,J) = (0.D0,0.D0)
      DCFC(I,J) = (0.D0,0.D0)
```

```
          DCFC1(I,J) = (0.D0,0.D0)
          FC(I,J) = (0.D0,0.D0)
          FCS(I,J) = (0.D0,0.D0)
          XVU(I,J) = (0.D0,0.D0)
          XVV(I,J) = (0.D0,0.D0)
          XOT(I,J) = (0.D0,0.D0)
          SXO(I,J) = (0.D0,0.D0)
          SXX(I,J) = (0.D0,0.D0)
          RU(I,J) = (0.D0,0.D0)
          RV(I,J) = (0.D0,0.D0)
          RVV(I,J) = (0.D0,0.D0)
          RVU(I,J) = (0.D0,0.D0)
          RX(I,J) = (0.D0,0.D0)
          RX2(I,J) = (0.D0,0.D0)
10    CONTINUE
          DO 12 I = 1,20
          WK(I) = 0.D0
          WK1(I) = 0.D0
          GC(I) = 0.
          WA(I) = 0.D0
          WK4(I) = 0.D0
12    WK2(I) = 0.D0
          DO 11 I = 1,N1
          DO 11 J = 1,M1
          BS1(I,J) = B(I,J)
11    BS(I,J) = B(I,J)
          DO 13 I = 1,250
          WAREA3(I) = 0.D0
          WAREA2(I) = 0.D0
13    WAREA1(I) = 0.D0
          DO 5 I = 1,100
          VLB(I) = 0.
          VUB(I) = 0.
5     XBJ(I) = 0.
          DO 14 I = 1,1000
14    IWK(I) = 0
          DO 15 I = 1,3000
15    WK3(I) = 0.0
C************************************************************************
C
C        REINIT UT, BUILD COMPLEX A  & FIND SVD OF B
C
C************************************************************************
          DO 20 I = 1,N1
          UNITY(I,I) = 1.D0
          CUT(I,I) = 1.D0
          UNITYC(I,I) = (1.D0,0.D0)
          UT(I,I) = 1.D0
          DO 20 J = 1,N1
20    AC(I,J) = DCMPLX(A(I,J),0.D0)
          CALL LSVDF(BS,10,N1,M1,UT,10,N1,S,WK,IER)
          CALL RANKD(BS,UT,S,N1,M1,U0TC,U1TC,R2INVC,IRANK,BGI)
          CALL FRTCMS('CLRSCRN ')
          IF(IER.NE.33)GOTO 25
          WRITE(6,24)IRANK
24    FORMAT(/5X,'**** B IS RANK DEFICIENT, RANK = ',I2,' ****',/)
          GOTO 42
C        DECOMPOSE B
C        FIND U
25    IRANK = M1
          DO 21 I = 1,N1
          DO 21 J = 1,N1
21    U(I,J) = UT(J,I)
C        FIND U0 & U0 TRANSPOSE
          DO 22 I = 1,N1
          DO 22 J = 1,M1
          U0(I,J) = U(I,J)
          U0C(I,J) = DCMPLX(U0(I,J),0.D0)
          U0T(J,I) = U(I,J)
```

```
22     UOTC(J.I) = DCMPLX(U(I.J).0.00)
C          FINO UI & U1 TRANSPOSE
             00 23 I = 1.N1
             00 23 J = M1 + I.N1
             UI(I.J) = U(I.J)
             UIT(J-M1.1) = U(I.J)
23     UITC(J-M1.1) = OCMPLX(U(I.J).0.D0)
             00 30 I = 1.M1
             00 30 J = 1.M1
             VI(I.J) = BS(I.J)
30     CONTINUE
C           WRITE U FOR OEBUG
C      WRITE(6.500)
C00    FORMAT(/5X.'**** U FCR B = UOV**T) ****'./)
C      00 550 J = 1.NI
C      WRITE(6.525)(U(J.K).K=1.N1)
525    FORMAT(1X.6(EI0.3.IX))
C50    CONTINUE
C         WRITE UOT ANO UIT FOR OEBUG
C      WRITE(6.600)
C00    FORMAT(/5X.'**** UOT ****'./)
C      00 650 J = 1.N1
C      WRITE(6.525)(UOT(J.K).K=1.N1)
C50    CONTINUE
C      WRITE(6.700)
C00    FORMAT(/5X.'**** U1T ****'./)
C      00 750 J = 1.N1
C      WRITE(6.525)(U1T(J.K).K=I.NI)
C50    CONTINUE
C************************************************************************
C
C         FINO RZ = OIAG<LAMO1 LAMO2 ....>V**T
C
C************************************************************************
             00 40 I = 1.M1
             00 40 J = I.M1
40     VT(I.J) = V1(J.I)
             00 50 1 = 1.M1
50     SM(I.I) = S(1)
             CALL VMULFF(SM.VT.M1.M1.M1.10.I0.RZ.10.1ER)
             00 41 I = 1.M1
             DO 41 J = 1.M1
41     RZC(I.J) = OCMPLX(RZ(I.J).0.00)
C************************************************************************
C
C         COMPUTE U*SIGMA*V TRANSPOSE FOR PGM DEBUG
C
C************************************************************************
             DO B25 J = I.N1
825    SMC(J.J) = S(J)
             CALL VMULFF(SMC.VT.N1.M1.M1.10.10.SV.I0.IER)
             CALL VMULFF(U.SV.N1.M1.MI.I0.10.BCHK.10.1ER)
             WR1TE(6.B00)
B00    FORMAT(/5X.'**** U * SIGMA * VTRAN ****'./)
             00 850 J = 1.N1
             WRITE(6.525)(8CHK(J.K).K=1.M1)
850    CONTINUE
C************************************************************************
C
C        "SELECT" EIGENVECTOR MATRIX
C         (PROGRAM EITHER REAOS 1N THE OESIREO SET FROM SYSTEM OATA
C          FILE OR WILL REAO LATEST DESIGN EIGENVECTOR MATRIX FROM
C          PREVIOUS COMPUTATION)
C************************************************************************
42     WRITE(6.69)
69     FORMAT(/10X.'**** PRESENTLY IN KVECT. ENTER "1" 1F YOU OESIRE ****
           1'./10X.     '**** TO USE PREV1OUSLY COMPUTEO EIGENVECTOR MA-  ****'
           2./10X.      '**** TRIX OR ENTER "2" IF YOU OESIRE TO RECOMPUTE****'
           3./10X.      '**** THE EIGENVECTOR MATRIX.                     ****'
```

```
          4./)
            READ(*,1050)IOPT
            IF(IOPT.EQ.2)GOTO 67
            DO 68 J = 1,N1
            DO 68 K = 1,N1
            READ(2,64)V(K,J)
64    FORMAT(2E12.5)
68    CONTINUE
            CALL FRTCMS('CLRSCRN ')
            GOTO 66
67    WRITE(6,70)
70    FORMAT(/5X,'**** PRESENTLY IN KVECT SUBROUTINE, BASE ****',/5X,
          1          '**** EIGENVECTORS ARE FROM DATA FILE.    ****',/5X,
          2          '**** PRESENTLY CHECKING IF THEY ARE MEM- ****',/5X,
          3          '**** BERS OF ALLOWABLE SUPSPACE.         ****',/)
51    DO 75 I = 1,N1
            DO 75 J = 1,N1
75    V(I,J) = VD(I,J)
C***********************************************************************
C
C       CHECK IF DESIRED BASE RIGHT HAND EIGENVECTORS ARE MEMBERS OF THE
C        NULL SPACE OF <U1 TRANSPOSE * <A - LAMDHA I * I>>
C
C***********************************************************************
            ICONJG = 0
            DO 55 I = 1,N1
            Z1 = EIGD(I)
            DO 56 J = 1,N1
            DO 56 K = 1,N1
56    AML(J,K) = DCMPLX(A(J,K),0.D0) - Z1*UNITYC(J,K)
            IF(IRANK.EQ.M1)GOTO 43
            IY = N1-IRANK
            GOTO 44
43    IY= N1 - M1
44    CALL CMATML(U1TC,AML,IY,N1,N1,RNULLC)
            DO 57 J = 1,N1
57    WV(J) = V(J,I)
59    CALL CMATML(RNULLC,WV,IY,N1,1,WV1C)
            DO 58 J = 1,IY
58    RES(J,I) = WV1C(J)
C        FIND 2-NORM OF THE RESIDUAL
            Y = 0.D0
            DO 76 J = 1,IY
76    Y = Y + DREAL(RES(J,I))**2+ DIMAG(RES(J,I))**2
            ROBJ = DSQRT(Y)
C        IF 2-NORM OF RESIDUAL IS OK -- JUMP OUT
            IF(ROBJ.LE.1.D-06)GOTO 55
C***********************************************************************
C
C        IF EIGENVALUE IS COMPLEX, THEN PERFORM THE OPTIMIZATION
C        ONE TIME FOR BOTH THE EIGENVALUE AND ITS CONJUGATE.
C
C***********************************************************************
            IF(DIMAG(Z1).EQ.0.D0)GOTO 960
            IF(ICONJG.EQ.1)GCTO 97
            ICONJG = 1
960   WRITE(6,1000)I,ROBJ
1000  FORMAT(///5X,'**** RESIDUAL TOLERANCE EXCEEDED FOR DESIRED   ****
          1',/5X,        '**** BASE EIGENVECTOR NO. ',I2,' ENTERING ADS
          2****',/5X,    '**** WITH ROBJ = ',E12.5,  '                    ****'
          3,/5X,         '**** C-LOOP EIGENVALUES HAVE BEEN SPECIFIED..  ****
          4',/5X,        '**** ENTER NO. OF ELEMENTS OF THE EIGENVECTOR  ****
          5',/5X,        '**** WHICH MUST BE ARBITRARY.                  ****
          6',/)
            READ(*,1050)IELE
            CALL FRTCMS('CLRSCRN ')
            IF(IELE.EQ.0)GOTO 962
961   WRITE(6,1001)IELE,I
1001  FORMAT(/5X,'**** ENTER ROW NUMBERS OF THE ',I2,' ELEMENTS    ****'
     1,/5X,    '***** OF EIGENVECTOR NO.',I2,' WHICH ARE ARBITRARY  ****'
     2,/5X,      '***** IN  I2 FORMAT.  EXAMPLE: "0102"             ****'
     3,/)
      READ(*,1002)(IROW(J,I),J=1,IELE)
1002  FORMAT(10I2)
      CALL FRTCMS('CLRSCRN ')
C**********************************************************************
C
C        SET UP FOR ADS OPTIMIZER CALL IN SINGLE PRECISION FOR RESIDUAL
C
C**********************************************************************
962   SROBJ = SNGL(ROBJ)
      INFO = 0
      ISTRAT = 0
      IOPT = 3
      IONED = 1
      NDV = 2*N1
      NCON = 0
      IGRAD = 0
      IPRINT = 1000
      NGT = 0
      NRA = 10
      NCOLA = 10
C        SET BOUNDS
C        ASK ALLOWED TOLERANCE ON DESIGN EIGENVECTORS
      IF(IELE.EQ.0)GOTO 1007
1008  WRITE(6,1003)
1003  FORMAT(/5X,'***** ENTER REAL UPPER BOUND OF ARBITRARY ELEMENTS ****
     1',/5X,   '***** IN "F12.5" FORMAT. ENTER "1" FOR NO BOUND.    ****')
      READ(*,1020)UR
      WRITE(6,1004)
1004  FORMAT(/5X,'***** ENTER IMAG UPPER BOUND OF ARBITRARY ELEMENTS ****
     1',/5X,   '***** IN "F12.5" FORMAT. ENTER "1" FOR NO BOUND.    ****')
      READ(*,1020)UC
      IF(UR.NE.1.E+06)GOTO 1006
      UR = 0.1E+21
1006  IF(UC.NE.1.E+06)GOTO 1005
      UC = 0.1E+21
1005  IF(IELE.EQ.N1)GOTO 1013
1007  WRITE(6,1010)
1010  FORMAT(/5X,'***** ENTER ALLOWED EIGENVECTOR TOLERANCE IN "F12.5" FO
     1RMAT ****',/5X,'***** FOR THOSE EIGENVECTOR ELEMENTS      ****',/5X,
     2                 '***** WHICH ARE NOT ARBITRARY.            ****',/)
      WRITE(6,1011)
1011  FORMAT(5X,'***** ENTER TOLERANCE ON REAL PART ****',/)
      READ(*,1020)EPSR
1020  FORMAT(F12.5)
      WRITE(6,1012)
1012  FORMAT(5X,'***** ENTER TOLERANCE ON IMAG PART ****',/)
      READ(*,1020)EPSC
1013  CALL FRTCMS('CLRSCRN ')
      JJ = 0
      DO 77 J = 1,NDV-1,2
      JJ = JJ + 1
      KFIND = 0
      IF(IELE.EQ.0)GOTO 1026
1014  DO 1025 K = 1,IELE
      IF(JJ.NE.IROW(K,I))GOTO 1025
      KFIND = 1
      VUB(J) = UR
      VUB(J+1) = UC
      VLB(J) = -UR
      VLB(J+1) = -UC
1025  CONTINUE
      IF(KFIND.EQ.1)GOTO 1027
1026  X = DREAL(VD(JJ,I))
      Y = DIMAG(VD(JJ,I))
      VUB(J) = SNGL(X) + EPSR
      VUB(J+1) = SNGL(Y) + EPSC
```

```
          VLB(J) = SNGL(X) - EPSR
          VLB(J+1) = SNGL(Y) - EPSC
1007  X1 = DREAL(WV(JJ))
          Y1 = DIMAG(WV(JJ))
          SWV(J) = SNGL(X1)
          SWV(J+1) = SNGL(Y1)
          IC(J) = 0
          IC(J+1) = 0
          DF(J) = 0
          DF(J+1) = 0
77    CONTINUE
84    CALL ADS(INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,SWV,VLB,
        1 VUB,SROBJ,G,IDG,NGT,IC,DF,RA,NRA,NCOLA,WK3,3000,IWK,1000)
          JJ = 0
          DO 78 J = 1,NDV-1,2
          JJ = JJ + 1
          WV(JJ) = CMPLX(SWV(J),SWV(J+1))
78    V(JJ,I) = WV(JJ)
          IF(INFO.EQ.0)GOTO 54
C        REEVALUATE OBJECTIVE FUNCTION
          CALL CMATML(RNULLC,WV,IY,N1,1,WVIC)
          DO 82 J = 1,IY
82    RES(J,I) = WVIC(J)
          Y = 0.D0
          DO 83 J = 1,IY
83    Y = Y + DREAL(RES(J,I))**2 + DIMAG(RES(J,I))**2
          ROBJ = DSQRT(Y)
          SROBJ = SNGL(ROBJ)
C        CONTINUE WITH ADS OPTIMIZATION (MINIMIZING ROBJ)
          GOTO 84
54    WRITE(6,95)ROBJ,I
95    FORMAT(/1X,'**** OBJECTIVE FUNCTION VALUE = ',E10.3,' ****',/1X,
        1             '**** FOR EIGENVECTOR NO. ',I2,'              ****',/)
          IF(DIMAG(Z1).EQ.0.D0)GOTO 55
          IF(ICONJG.EQ.1)GOTO 55
97    DO 98 K = 1,N1
          ICONJG = 0
98    V(K,I) = DCONJG(V(K,I-1))
55    CONTINUE
C********************************************************************
C
C        DISPLAY RESULTS OF <U1T*<A-LAMDHA*I>>*DESIRED BASE EIGENVECTORS
C
C********************************************************************
          WRITE(6,65)
65    FORMAT(//5X,'**** THE FOLLOWING MATRIX DISPLAYS THE "NEARNESS" OF*
        1***',/5X,  '**** THE DESIGN EIGENVECTORS TO THE ALLOWABLE SUB-   *
        2***',/5X,  '**** SPACE FOR THE SYSTEM UNDER ANALYSIS IN COLUMN   *
        3***',/5X,  '**** FORMAT. IF THIS "NEARNESS" MATRIX IS NOT SATIS- *
        4***',/5X,  '**** FACTORY, ENTER "1". KVECT WILL NOW USE THE DES- *
        5***',/5X,  '**** IGN EIGENVECTORS AS THE NEW DESIRED BASE EIGEN- *
        6***',/5X,  '**** VECTORS AND REPEAT OPTIMIZER ROUTINE. OTHERWISE *
        7***',/5X,  '**** ENTER "0". NOTE: "ZERO" = 1.0-06                *
        8***',/)
          DO 63 I = 1,IY
          WRITE(6,61)(RES(I,J),J=1,N1)
61    FORMAT(1X,8(E8.1,1X))
63    CONTINUE
          READ(*,1030)ISAT
1030  FORMAT(I1)
          IF(ISAT.EQ.0)GOTO 66
          DO 93 J = 1,N1
          DO 93 K = 1,N1
93    VD(J,K) = V(J,K)
          GOTO 51
C********************************************************************
C
C        FIND INV(V)  .... INVERSE OF OPTIMIZED EIGENVECTORS
C       TO CHK FOR INVERTIBILITY
```

```
C
C**********************************************************************
66    DO 94 J = 1.N1
            DO 94 K = 1.N1
            IF(IOPT.EQ.1)GOTO 211
            WRITE(2.64)V(K.J)
211   VD(J.K) = V(J.K)
94    VSAVE(J.K) = V(J.K)
C          DEBUG WRITE STATEMENTS
C     TITLE='V'
C     CALL CWRITE(V.N1.N1,TITLE)
C     CALL CWRITE(UNITYC.N1.N1.TITLE)
            CALL LEQT1C(VSAVE.N1.10.UNITYC.N1,10.0.WA.IER)
C     CALL CWRITE(UNITYC.N1.N1.TITLE)
C     CALL CMATML(UNITYC.V.N1.N1.N1.VIV)
C     CALL CWRITE(VIV.N1.N1.TITLE)
C        RESTORE UNITYC AND REPLACE WITH VDI
            DO 210 I = 1.N1
            DO 210 J = 1.N1
            VDI(I.J) = UNITYC(I.J)
210   UNITYC(I.J) = DCMPLX(UNITY(I.J).0.D0)
C**********************************************************************
C
C        FIND 2-NORM CONDITION OF OPTIMIZED EIGENVECTOR MATRIX
C
C**********************************************************************
            DO 79 I = 1.N1
            DO 79 J = 1.N1
            VDS(I.J) = V(I.J)
79    VDIS(I.J) = VDI(I.J)
            CALL CSVD(VDS.10.10.N1.N1.0.N1.N1,S1.UVDS.VVDS)
            CALL CSVD(VDIS.10.10.N1.N1.0.N1.N1.S2.UVDIS.VVDIS)
            COND2 = S1(1)*S2(I)
C     CALL FRTCMS('CLRSCRN ')
92      WRITE(6.85)COND2
85      FORMAT(/5X,'**** 2-NORM CONDITION NO. OF EIGENVECTOR MATRIX ****
          1'./5X,'**** IS '.E12.5.'     ****'./)
            WRITE(6.1040)
1040  FORMAT(//5X.'**** IF 2-NORM COND NO. IS UNSAT:              ****'./5X
          1.            '**** ENTER "1" TO MIN COND<V> VIA ADS.        ****'./5X
          2.            '**** ENTER "2" TO RETURN TO ADS MIN OF RE-    ****'./5X
          3.            '****                SIDUAL OF <V>.            ****'./5X
          4.            '**** IF 2-NORM COND NO. IS SAT: ENTER"3"      ****'./5X
          5./)
            READ(*.1050)ICOND
1050  FORMAT(I1)
            CALL FRTCMS('CLRSCRN ')
            IF(ICOND.EQ.2)GOTO 51
86    IF(ICOND.EQ.3)GOTO 91
C**********************************************************************
C
C        SET UP OPTIMIZER CALL FOR 2-NORM CONDITION NO. OF V
C        NOCON = # OF ORTHOGONALITY CONSTRAINTS
C        NSSCON = # OF ALLOWABLE SUBSPACE CONSTRAINTS
C        NCON = TOTAL # OF CONSTRAINTS
C   CAUTION!!!  NOT OPERATIONAL
C
C**********************************************************************
            COBJ = SNGL(COND2)
            INFO = 0
C**********************************************************************
            ISTRAT = 0
            IOPT = 4
            IONED = 7
C**********************************************************************
            NSSCON = N1
C     WRITE(6.1070)
C070  FORMAT(//5X.'**** ENTER NUMBER OF ORTHOGONALITY CONSTRAINTS ****'.
C    1 /5X.      '**** IN I2 FORMAT. EXAMPLE: "06"              ****'./
C     2 )
C     READ(*,1002)NOCON
            NCON =  NSSCON
            IGRAD = 0
            IPRINT = 2020
            NGT = 0
            NRA = 10
            NCOLA = 10
            NDV = 2*N1*N1
            IY = N1 - M1
C        IF THIS IS NOT THE FIRST PASS THROUGH THE CONDITION NUMBER
C        MINIMIZATION. ASK THE USER IF HE/SHE DESIRES TO USE THE
C        CURRENT EIGENVECTOR MATRIX OR IF THEY DESIRE TO USE THE
C        EIGENVECTOR MATRIX COMPUTED DURING THE RESIDUAL MINIMIZATION.
            WRITE(6,1071)
1071  FORMAT(/5X,'**** IF THIS IS THE INITIAL CONDITION NUMBER PASS ****
     1',/5X,     '**** ENTER "2". IF NOT AND YOU DESIRE TO REENTER  ****
     2',/5X,     '**** THE OPTIMIZATION WITH THE UPDATED EIGENVECTOR****
     3',/5X,     '**** ENTER "2". OTHERWISE ENTER "1" WHICH WILL RE-****
     4',/5X,     '**** START THE OPTIMZATION WITH THE EIGENVECTOR   ****
     5',/5X,     '**** MATRIX IN "DESVEC" DATA FILE.                ****
     6',/)
            READ(*,1050)IICON
            IF(IICON.EQ.2)GOTO 1072
            DO 1073 J = 1,N1
            DO 1073 K = 1,N1
1073  V(J,K) = VD(J,K)
C        ASK ALLOWED TOLERANCE ON EIGENVECTOR ELEMENTS
1072  WRITE(6,1080)
1080  FORMAT(/5X,'**** AT LEAST TWO EIGENVECTORS MUST FLOAT DURING THIS
     1 ****',/5X,'**** OPTIMIZATION. ENTER THE COLUMN NO.S OF THE EIGEN-
     2 ****',/5X,'**** VECTORS WHICH MAY "FLOAT" IN ASCENDING ORDER IN
     3 ****',/5X,'**** I2 FORMAT. EXAMPLE: "0102" WILL FLOAT EV # 1 & 2.
     4 ****',/)
            READ(*,1002)(IFLT(I),I=1,N1)
            WRITE(6,1003)
            READ(*,1020)UR
            WRITE(6,1004)
            READ(*,1020)UC
            IF(UR.NE.1.E+06)GOTO 1082
1081  UR = 0.1E+21
1082  IF(UC.NE.1.E+06)GOTO 1084
1083  UC = 0.1E+21
1084  WRITE(6,1010)
            WRITE(6,1011)
            READ(*,1020)EPSR
            WRITE(6,1012)
            READ(*,1020)EPSC
            J = 1
C        REINIT AND SET BOUNDS
            DO 130 K = 1,N1
            IC1(K) = 0
            DF(K) = 0.
            DO 130 L = 1,N1
            X = DREAL(V(L,K))
            Y = DIMAG(V(L,K))
            XBJ(J) = SNGL(X)
            XBJ(J+1) = SNGL(Y)
            DO 131 II = 1,N1
131   IF(IFLT(II).EQ.K)GOTO 200
            VUB(J) = XBJ(J) + EPSR
            VUB(J+1) = XBJ(J+1) + EPSC
            VLB(J) = XBJ(J) - EPSR
            VLB(J+1) = XBJ(J+1) - EPSC
            J = J + 2
            GOTO 130
200   VUB(J) = UR
            VUB(J+1) = UC
            VLB(J) = -UR
            VLB(J+1) = -UC
            J = J + 2
130   CONTINUE
C        DEBUG WRITE STATEMENT
C     WRITE(6,2000)
C000  FORMAT(/5X,'**** VECTOR BOUNDS FOR ADS ****',/)
C     DO 2001 J = 1,NDV
C     WRITE(6,2002)VLB(J),XBJ(J),VUB(J)
C002  FORMAT(1X,3(E12.5,2X))
C001  CONTINUE
C        DEFINE CONSTRAINTS AS NONLINEAR, INEQUALITY
            DO 135 I = 1,NCON
135   IDG1(I) = 0
C        SET BOUNDS ON CONSTRAINTS
C     WRITE(6,1060)
C060  FORMAT(/5X,'**** ENTER ALLOWED TOLERANCE ON NON-ORTHOGONALITY ****
C    1',/5X,      '****        OF EIGENVECTORS IN  DECIMAL FORMAT.    ****
C    2',/)
C     READ(*,1020)E
            WRITE(6,1062)
1062  FORMAT(/5X,'**** ENTER NEW DESIRED EIGENVALUES FOR CONDITION  ****
          1',/5X,     '**** NO. MIN. ENTER IN SAME ORDER AS DATA FILE.   ****
          2',/5X,     '**** NOTE: EIGENVALUES MUST BE ENTERED IN SAME    ****
          3',/5X,     '**** ORDER AS CORRESPONDING EIGENVECTOR. IF EI-   ****
          4',/5X,     '**** GENVALUE CORRESPONDS TO A "NON-FLOATING"     ****
          5',/5X,     '**** EIGENVECTOR, IT SHOULD REMAIN THE SAME AS    ****
          6',/5X,     '**** ITS INPUT VALUE TO MAINTAIN DESIGN RQMTS.    ****
          7',/)
            DO 1063 J = 1,N1
            WRITE(6,1064)J
1064  FORMAT(/10X,'**** EIGENVALUE NO. ',I2,' ****',/5X,'REAL PART =',/)
            READ(*,1065)X
1065  FORMAT(F12.5)
            WRITE(6,1066)
1066  FORMAT(5X,'IMAG PART =',/)
            READ(*,1065)Y
            EIGD(J) = DCMPLX(X,Y)
1063  CONTINUE
            WRITE(6,1061)
1061  FORMAT(/5X,'**** ENTER ALLOWED TOLERANCE ON ACHIEVABLE SUB-    ****
          1',/5X,     '**** SPACE CONSTRAINTS. THIS VALUE CORRESPONDS    ****
          2',/5X,     '**** TO THE ALLOWABLE EUCLIDEAN DISTANCE BETWEEN  ****
          3',/5X,     '**** THE DESIRED EIGENVECTOR AND THE ACHIEVABLE   ****
          4',/5X,     '**** SUBSPACE CORRESPONDING TO THE EIGENVALUE EN- ****
          5',/5X,     '**** TERED ABOVE.                                 ****
          6',/)
            READ(*,1020)E1
            CALL FRTCMS ('CLRSCRN ')
C        OPTIMIZE
190   CALL ADS(INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,XBJ,
          1 VLB,VUB,COBJ,GC,IDG1,NGT,IC1,OF,RA,NRA,NCOLA,WK3,3000,IWK,1000)
C        EVALUATE OBJECTIVE & CONSTRAINTS
            L = 1
            DO 140 J = 1,N1
            DO 140 K = 1,N1
            V(K,J) = CMPLX(XBJ(L),XBJ(L+1))
            VSAVE(K,J) = V(K,J)
            L = L + 2
140   CONTINUE
            CALL LEQT1C(VSAVE,N1,10,UNITYC,N1,10,0,WA,IER)
            DO 142 J = 1,N1
            DO 142 K = 1,N1
142   VDI(J,K) = UNITYC(J,K)
            DO 145 J = 1,N1
            DO 145 K = 1,N1
145   UNITYC(J,K) = DCMPLX(UNITY(J,K),0.D0)
            CALL CNSTRN(N1,GC,XBJ,E,E1,NOCON,A,B,U1TC,EIGO,UNITY,IY,UNITYC
          1 ,IFLT)
            DO 150 J = 1,N1
          DO 150 K = 1,N1
          VDS(J,K) = V(J,K)
150   VD1S(J,K) = VD1(J,K)
          CALL CSVD(VDS,10,10,N1,N1,0,N1,N1,S1,UVDS,VVDS)
          CALL CSVD(VD1S,10,10,N1,N1,0,N1,N1,S2,UVD1S,VVD1S)
          COND2 = S1(1)*S2(1)
C     CALL NORM(V,N1)
C     CALL NORM(VD1,N1)
          IF(INFO.EQ.0)GOTO 92
          COBJ = SNGL(COND2)
          GOTO 190
C*********************************************************************
C
C        FIND <V><SM><VD1> **** NOTE: VD1 = INV<V>
C
C*********************************************************************
91    DO 80 I = 1,N1
80    EIGDM(I,I) = EIGD(I)
          CALL CMATML(EIGDM,VDI,NI,N1,N1,EVI)
          CALL CMATML(V,EVI,N1,N1,N1,VEVI)
          DO 90 I = 1,N1
          DO 90 J = 1,N1
90    VMA(I,J) = VEVI(I,J) - DCMPLX(A(I,J),0.D0)
C        FIND SVD OF C
          DO 231 J = 1,L1
          DO 231 K = 1,N1
          CSS(J,K) = C(J,K)
231   CS(J,K) = C(J,K)
          CALL LSVDF(CS,10,L1,N1,CUT,10,L1,SC,WK4,IER)
          CALL RANKD(CS,CUT,SC,L1,N1,C0TC,C1TC,ZINVCC,IRANKC,CG1)
          WRITE(6,235)IRANKC
235   FORMAT(/5X,'**** C MATRIX HAS RANK = ',I2,' ****',/)
          IF(IRANK.EQ.M1.AND.IRANKC.EQ.N1)GOTO 229
          IY = N1 - IRANK
          CALL FEEDEF(A,B,C,V,VDI,EIGDM,BGI,U1TC,CGI,N1,MI,LI,IY,IFEED,FC)
          GOTO 238
229   CALL CMATML(U0TC,VMA,M1,N1,N1,UTVC)
C*********************************************************************
C
C        FIND INVERSE<RZ> & FEEDBACK GAINS
C
C*********************************************************************
224   CALL LINV2F(RZ,M1,10,RZINV,1,WAREA2,IER)
          DO 96 J = 1,MI
          DO 96 K = 1,M1
96    RZINVC(J,K) = DCMPLX(RZINV(J,K),0.D0)
          CALL CMATML(RZINVC,UTVC,M1,M1,N1,FC)
C        MODIFY FC IF OTHER THAN FULL STATE FEEDBACK
          IF(IFEED.EQ.1)GOTO 228
237   CALL CMATML(FC,ZINVCC,M1,N1,IRANKC,FCZ)
          CALL CMATML(FCZ,C0TC,M1,IRANKC,L1,FC)
238   IF(IFEED.EQ.2)GOTO 228
          DO 232 J = 1,L1
          DO 232 K = 1,M1
          FCS(K,J) = FC(K,J)
232   DC(J,K) = DCMPLX(D(J,K),0.D0)
          CALL CMATML(DC,FC,L1,M1,L1,DCFC)
          DO 233 J = 1,L1
          DO 233 K = 1,L1
233   DCFCI(J,K) = UNITYC(J,K) + DCFC(J,K)
          CALL LEQT1C(DCFCI,L1,10,UNITYC,L1,10,0,WA,IER)
          CALL CMATML(FCS,UNITYC,M1,L1,L1,FC)
          DO 234 J = 1,N1
          DO 234 K = 1,N1
234   UNITYC(J,K) = DCMPLX(UNITY(J,K),0.D0)
C*********************************************************************
C
C        FLAG USER IF COMPLEX FEEDBACK GAINS ARE COMPUTED
C
```

```
C*******************************************************************************
228   DO 220 J = 1,M1
          DO 220 K = 1,L1
          TIMAG = DIMAG(FC(J,K))
          IF(DABS(TIMAG).GT.1.0-06)GOTO 221
          GOTO 220
221   X = DIMAG(FC(J,K))
          WRITE(6,222)J,K,X
222   FORMAT(/5X,'**** WARNING! F(',I2,',',I2,') HAS AN      ****',/5X,
        1            '**** IMAGINARY PART = ',E12.5,/5X,
        2            '**** EIGENVECTOR MATRIX MAY BE ILL CONDITIONED ****',/
        3)
220   CONTINUE
C*******************************************************************************
C
C        SORT OUT REAL PARTS OF F MATRIX
C
C*******************************************************************************
          DO 132 J = 1,M1
        . DO 132 K = 1,N1
132   F(J,K) = DREAL(FC(J,K))
C     WRITE(6,100)
C00   FORMAT(/5X,'**** F MATRIX RESULTS FROM KAUTSKY ALGORITHM ****',/)
C     DO 110 I = 1,M1
C     WRITE(6,120)(F(I,J),J= 1,N1)
C20   FORMAT(1X,6(E10.3,1X))
C10   CONTINUE
C        FIND MINIMUM SINGULAR VALUE IN FREQUENCY RANGE .01 - 100 R/S
          CALL SVA(A,B,C,D,F,UNITY,N1,M1,L1,W,SVM,SVMAX,SVM1,SVMAX1,IFEED)
          X1 = 20.D0
          DO 230 I = 1,200
          IF(SVM(I).GT.X1)GOTO 230
          SMIN = SVM(I)
          X2 = W(I)
          X1 = SMIN
230   CONTINUE
          WRITE(6,240)SMIN,X2
240   FORMAT(/5X,'**** MIN S.V. OF RDM = ',E12.5,' ****',/5X,
        1            '**** FREQ. OF MIN. S.V. = ',E12.5,' ****',/5X,
        2            '**** ENTER "1" TO OPTIMIZE THE S.V. ****',/5X,
        3            '**** ENTER "2" TO EXIT KVECT.       ****',/)
          READ(*,1050)ILOG
          IF(ILOG.EQ.2)GOTO 133
C*******************************************************************************
C
C        SET UP OPTIMIZER CALL FOR MINIMIZING MULTI-VARIABLE
C        KALMAN INEQUALITY FOR R = IDENTITY. ONLY BUILD ONE
C        DESIGN VARIABLE VECTOR AND UPPER/LOWER BOUND VECTOR
C        PER COMPLEX EIGENVALUE: PRESENTLY ONLY REAL CASES ARE HANDLED
C
C*******************************************************************************
          CALL FRTCMS('CLRSCRN ')
          WRITE(6,245)
245   FORMAT(/5X,'**** ENTER NO. OF ITERATIONS DESIRED FOR ADS ****',/)
          READ(*,1002)NITER
          CALL FRTCMS('CLRSCRN ')
          WRITE(6,246)
246   FORMAT(/5X,'**** ENTER MIN. S.V. TOLERANCE FOR ADS ****',/)
          READ(*,1020)SVTOL
          CALL FRTCMS('CLRSCRN ')
          INFO = 0
          ISTRAT = 1
          IOPT = 1
          IONED = 2
          NCON = N1
          IGRAD = 0
          IPRINT = 2200
          NGT = 0
          NRA = 10
          NCOLA = 10
          NDV = 2*N1*N1
          IY = N1 - M1
          ITER = 0
C        SPECIFY EIGENSTRUCTURE
C        NORMALIZE EIGENVECTOR MATRIX FIRST
          CALL NORM(V,N1)
          DO 250 I = 1,N1
          WRITE(6,255)I
255   FORMAT(//5X,'**** ENTER NO. OF ELEMENTS OF EIGENVECTOR ',I2,' ****
     1',/5X,      '**** WHICH ARE ARBITRARY.                        ****',/)
250   READ(*,1050)IELEM(I)
          CALL FRTCMS('CLRSCRN ')
          DO 260 I = 1,N1
          WRITE(6,1001)IELEM(I),I
260   READ(*,1002)(IROW(J,I),J=1,IELEM(I))
          CALL FRTCMS('CLRSCRN ')
C        SET BOUNDS
          WRITE(6,1003)
          READ(*,1020)UR
          WRITE(6,1004)
          READ(*,1020)UC
          CALL FRTCMS('CLRSCRN ')
          IF(UR.NE.1.E+06)GOTO 300
          UR = 0.1E+21
300   IF(UC.NE.1.E+06)GOTO 310
          UC = 0.1E+21
310   WRITE(6,1010)
          WRITE(6,1011)
          READ(*,1020)EPSR
          WRITE(6,1012)
          READ(*,1020)EPSC
          CALL FRTCMS('CLRSCRN ')
C     WRITE(6,319)UR,UC,EPSR,EPSC
CI9   FORMAT(/5X,'UR = ',E12.5,/5X,'UC = ',E12.5,/5X,'EPSR = ',E12.5,
C    I /5X,'EPSC = ',E12.5)
          J = 1
          DO 330 II = 1,N1
          DO 330 JJ = 1,N1
          DO 320 KK = 1,IELEM(II)
          IF(JJ.EQ.IROW(KK,II))GOTO 325
320   CONTINUE
          GOTO 340
325   VUB(J) = UR
          VUB(J+1) = UC
          VLB(J) = -UR
          VLB(J+1) = -UC
          J = J + 2
          GOTO 330
340   X = DREAL(V(JJ,II))
          Y = DIMAG(V(JJ,II))
          VUB(J) = SNGL(X) + EPSR
          VUB(J+1) = SNGL(Y) + EPSC
          VLB(J) = SNGL(X) - EPSR
          VLB(J+1) = SNGL(Y) - EPSC
          J = J + 2
330   CONTINUE
C        DEFINE CONSTRAINTS AS LINEAR INEQUALITY
          DO 335 I = 1,NCON
335   IDG1(I) = 2
C        SET BOUNDS ON CONSTRAINTS
          WRITE(6,1061)
          READ(*,1020)E1
          CALL FRTCMS('CLRSCRN ')
C        FILL - UR DESIGN VARIABLE VECTOR
331   JJ = 1
          DO 345 II = 1,N1
          DO 345 KK = 1,N1
          X = DREAL(V(KK,II))
```

```
            Y = DIMAG(V(KK.II))
            XBJ(JJ) = SNGL(X)
            XBJ(JJ+1) = SNGL(Y)
345     JJ = JJ + 2
C          DEBUG WRITE
            IF(INFO.NE.0)GOTO 349
            WRITE(6.348)
348     FORMAT(/5X,'**** ENTERING ADS WITH THE FOLLOWING ****'./)
            DO 346 I = 1.NDV
            WRITE(6.347)VLB(I).XBJ(I).VUB(I)
347     FORMAT(10X.3(E12.5.3X))
346     CONTINUE
C          OPTIMIZE
349     CALL ADS(INFO.ISTRAT.IOPT.IONED.IPRINT.IGRAD.NDV.NCON.XBJ.
          1 VLB.VUB.COBJ.GC.IDG1.NGT.IC1.DF.RA.NRA.NCOLA.WK3.3000.IWK.1000)
C          EVALUATE OBJECTIVE FUNCTION AND CONSTRAINTS
            ITER = ITER + 1
            JJ = 1
            ICONJ = 0
            DO 350 I = 1.N1
            DO 350 J = 1.N1
352     V(J.I) = CMPLX(XBJ(JJ).XBJ(JJ+1))
350     JJ = JJ + 2
C          NORMALIZE EIGENVECTOR MATRIX
            CALL NORM(V.N1)
            DO 351 I = 1.N1
            DO 351 J = 1.N1
351     VSAVE(I.J) = V(I.J)
            CALL LEQT1C(VSAVE.N1.10.UNITYC.N1.10.0.WA.IER)
            CALL CMATML(EIGCM.UNITYC.N1.N1.N1.EVI)
            CALL CMATML(V.EVI.N1.N1.N1.VEVI)
            DO 355 I = 1.N1
            DO 355 J = 1.N1
            VDI(I.J) = UNITYC(I.J)
            UNITYC(I.J) = DCMPLX(UNITY(I.J).0.D0)
355     VMA(I.J) =-DCMPLX(A(I.J),0.D0) + VEVI(I.J)
            CALL CMATML(UOTC.VMA.M1.N1.N1.UTVC)
            CALL CMATML(RZINVC.UTVC.M1.M1.N1.FC)
            TITLE = 'EV'
C       CALL CWRITE(V.N1.N1.TITLE)
C       CALL CWRITE(VDI.N1.N1.TITLE)
            DO 360 I = 1.M1
            DO 360 J = 1.N1
360     F(I.J) = DREAL(FC(I.J))
            CALL CNSTRN(N1.GC.XBJ.E.E1.NOCON.A.B.U1TC.EIGD.UNITY.IY.UNITYC,
          1 IFLT)
            CALL SVA(A.B.C.F.UNITY.N1.M1.L1.W.SVM.SVMAX.SVM1.SVMAX1)
            X1 = 20.D0
            DO 365 I = 1.200
            IF(SVM(I).GT.X1)GOTO 365
            SMIN = SVM(I)
            X2 = W(I)
            X1 = SMIN
365     CONTINUE
            COBJ = SNGL(SMIN) - 1.0
            CALL FRTCMS('CLRSCRN ')
            WRITE(6.370)COBJ.SMIN.X2.GC(1).GC(2).GC(3).GC(4)
370     FORMAT(/5X,'**** OBJ = '.E12.5.' ****'./5X,
          1            '**** MIN S.V. = '.E12.5.' AT FREO = '.E12.5./5X,
          2            '**** CONSTRAINT 1 = '.E12.5./5X.'**** CONSTRAINT 2 = '
          3.E12.5./5X.'**** CONSTRAINT 3 = '.E12.5./5X.'**** CONSTRAINT 4 = '
          4.E12.5./)
            IF(INFO.EO.0.OR.ABS(COBJ).LT.SNGL(SVTOL))GOTO 91
            IF(ITER.GT.NITER)GOTO 91
366     GOTO 349
C          WRITE RESULTS OF KVECT FOR FURTHER USE IN RECONFIGURATION
C          STUDIES.
133     DO 400 I = 1.N1
            WRITE(9.401)(V(I.J).J=1.N1)
```

```
401   FORMAT(1X,4E12.5,/1X,4E12.5)
400   CONTINUE
             DO 410 I = 1,N1
410   WRITE(9,401)(EIGDM(I,J),J=1,N1)
             DO 415 I = 1,M1
415   WRITE(9,401)(RZC(I,J),J=1,M1)
             DO 420 I = 1,N1
420   WRITE(9,401)(UOC(I,J),J=1,M1)
             DO 430 I = 1,M1
430   WRITE(9,401)(FC(I,J),J=1,N1)
             DO 435 I = 1,M1
435   WRITE(9,401)(F(I,J),J=1,N1)
             RETURN
             END
C**********************************************************************
C
C        SUBROUTINE CNSTRN: COMPUTES LINEAR, INEQUALITY CONSTRAINTS
C                           FOR OPTIMIZING THE 2-NORM CONDITION NO.
C                           OF THE DESIGN EIGENVECTOR MATRIX AND/OR
C                           MINIMIZING THE MULTI-VARIABLE KALMAN
C                           INEQUALITY RELATION.
C
C**********************************************************************
             SUBROUTINE CNSTRN(N1,G,X,E,E1,NOCON,A,B,U1TC,EIGD,UNITY,
           I  IY,UNITYC,IFLT)
             IMPLICIT REAL*8(A-H,P-Z)
             REAL*8 A(10,10),B(10,10),UNITY(10,10)
             COMPLEX*16 UNITYC(10,10),U1TC(10,10),EIGD(10),WM1(10,10),
           I RN(10,10),WV1(10),WV2(10),Z1
             COMPLEX*8 VX(200)
             REAL*4 G(20),X(100),E,E1,E2
             INTEGER N1,NOCON,IY,IFLT(10),NDV,INDEX
             NDV = 2*N1*N1
C     J = 1
C     DO 5 I = 1,NDV-1,2
C     VX(J) = CMPLX(X(I),X(I+1))
C     J = J + 1
C     IF(N1.EQ.2)GOTO 10
C     IF(N1.EQ.3)GOTO 20
C     IF(N1.EQ.4)GOTO 30
C     IF(N1.EQ.5)GOTO 1000
C     IF(N1.EQ.6)GOTO 1000
C     IF(N1.EQ.7)GOTO 1000
C     IF(N1.EQ.8)GOTO 1000
C**********************************************************************
C
C             ORTHOGONALITY CONSTRAINTS
C
C**********************************************************************
C0    G(1) = CABS(VX(1)*VX(3) + VX(2)*VX(4)) -E
C     GOTO 1000
C0    G(1) = CABS(VX(1)*VX(4) + VX(2)*VX(5) + VX(3)*VX(6))- E
C     G(2) = CABS(VX(1)*VX(7) + VX(1)*VX(8) + VX(1)*VX(9))- E
C     G(3) = CABS(VX(4)*VX(7) + VX(5)*VX(8) + VX(6)*VX(9))- E
C     GOTO 1000
C0    G(1) = CABS(VX(1)*VX(5) + VX(2)*VX(6) + VX(3)*VX(7) + VX(4)*VX(8)
C    1           ) - E
C     G(2) = CABS(VX(1)*VX(9) + VX(2)*VX(10) + VX(3)*VX(11) +
C    1                VX(4)*VX(12)) - E
C     G(3) = CABS(VX(1)*VX(13) + VX(2)*VX(14) + VX(3)*VX(15) +
C    1                VX(4)*VX(16)) - E
C     G(4) = CABS(VX(5)*VX(9) + VX(6)*VX(10) + VX(7)*VX(11) +
C    1                VX(8)*VX(12)) - E
C     G(5) = CABS(VX(5)*VX(13) + VX(6)*VX(14) + VX(7)*VX(15) +
C    1                VX(8)*VX(16)) - E
C     G(6) = CABS(VX(9)*VX(13) + VX(10)*VX(14) + VX(11)*VX(15) +
C    1                VX(12)*VX(16)) - E
C**********************************************************************
C
```

```
C              ALLOWABLE SUBSPACE CONSTRAINTS
C
C**********************************************************************
1000  LL = 1
         KK = 1
         DO 31 I = 1,N1
         Z1 = EIGD(I)
         DO 32 J = 1,N1
         DO 32 K = 1,N1
32    WM1(J,K) = DCMPLX(A(J,K),0.D0) - Z1*UNITYC(J,K)
         CALL CMATML(U1TC,WM1,IY,N1,N1,RN)
         JJ = 1
         INDEX = I * N1 * 2
         DO 35 LL = KK, INDEX-1 , 2
         WV1(JJ) = CMPLX(X(LL),X(LL+1))
         JJ = JJ + 1
35    CONTINUE
         KK = INDEX + 1
         CALL CMATML(RN,WV1,IY,N1,1,WV2)
         Y1 = 0.D0
         DO 36 II = 1,IY
36    Y1 = Y1 + CDABS(WV2(II))
         G(I) = SNGL(Y1) - E1
31    CONTINUE
         RETURN
         END
C**********************************************************************
C
C        SUBROUTINE CCNTRL: DETERMINES CONTROLLABILITY OF COMPLEX
C                             O-LOOP EIGENVALUES
C
C**********************************************************************
         SUBROUTINE CCNTRL(A,B,UNITYC,N1,M1,ICTR,EIG,I1)
         IMPLICIT REAL*8(A-H,P-Z)
         REAL*8 S(10),A(10,10),B(10,10)
         COMPLEX*16 UNITYC(10,10),WMC1(10,10),WMC2(10,10),WMC3(10,10),
       1EIG(10),WMC4(10,10),WMC5(20,20),Z1,Z2,Z3,U(10,10),V(10,10)
         INTEGER N1,M1,ICTR(10),I1
         Z1 = EIG(I1)
         DO 10 J = 1,N1
10    WMC1(J,J) = Z1*UNITYC(J,J)
         DO 20 J = 1,N1
         DO 20 K = 1,N1
         WMC2(J,K) = WMC1(J,K) - DCMPLX(A(J,K),0.D0)
20    WMC3(J,K) = UNITYC(J,K)
C        CALCULATE SLAMDHA (I)
         DO 30 J = 1,N1
         DO 30 K = 1,N1
30    WMC5(J,K) = WMC2(J,K)
         DO 40 J = 1,N1
         DO 40 K = 1,M1
40    WMC5(J,K+N1) = DCMPLX(B(J,K),0.D0)
C        CALCULATE RANK OF SLAMOHA
         II = M1+ N1
         CALL CSVD(WMC5,10,10,N1,II,0,N1,II,S,U,V)
         IF(S(N1).LE.0.D0)GOTO 50
         ICTR(I1) = 1
         GOTO 1000
50    ICTR(I1) = 0
1000  RETURN
         END
C**********************************************************************
C
C        SUBROUTINE CWRITE: WRITES OUT A COMPLEX MATRIX FOR
C                             IBM 327B TERMINAL
C                 A = INPUT COMPLEX MATRIX
C              IRA=ROWS OF A : IRC=COLUMNS OF A
C
C**********************************************************************
```

```
C     SUBROUTINE CWRITE(A,IRA,ICA,TITLE)
C     IMPLICIT REAL*8(A-H,P-Z)
C     COMPLEX*16 A(10,10)
C     INTEGER IRA,ICA
C     CHARACTER*80 TITLE
C     WRITE(6,5)
C     FORMAT(/,'***************************************************',/)
C     WRITE(6,6)TITLE
C     FORMAT(/,1A4)
C     DO 10 J = 1,IRA
C     WRITE(6,20)(A(J,K),K = 1,ICA)
C0    FORMAT(8(E8.1,1X))
C0    CONTINUE
C     WRITE(6,5)
C     RETURN
C     END
C************************************************************************
C
C             SUBROUTINE COND: FINDS 2 - NORM CONDITION NUMBER OF
C                               A SQUARE COMPLEX MATRIX
C
C************************************************************************
            SUBROUTINE CCND(Z,N1,XCOND)
            IMPLICIT REAL*8(A-H,P-Z)
            COMPLEX*16 Z(10,10),UZ(10,10),VZ(10,10),UNITYC(10,10),ZSAVE(10,10)
          1,UIZ(10,10),VIZ(10,10),Z1
            REAL*8 S1(10),S2(10),WA(10)
            INTEGER N1
C              INITIALIZE MATRICES
            DO 5 I = 1,10
            S1(I)= 0.D0
            S2(I) = 0.D0
            WA(I) = 0.D0
            DO 5 J = 1,10
            Z1 = (0.D0,0.D0)
            UZ(I,J) = Z1
            VZ(I,J) = Z1
            UNITYC(I,J) = Z1
            ZSAVE(I,J) = Z1
            UIZ(I,J) = Z1
            VIZ(I,J) = Z1
5     CONTINUE
            DO 10 I = 1,N1
            UNITYC(I,I) = DCMPLX(1.D0,0.D0)
            DO 10 J = 1,N1
            ZSAVE(I,J) = Z(I,J)
10    CONTINUE
            CALL CSVD(ZSAVE,10,10,N1,N1,0,N1,N1,S1,UZ,VZ)
            XCOND = S1(1)/S1(N1)
            RETURN
            END
C************************************************************************
C
C        SUBROUTINE NORM: NORMALIZES THE COLUMNS OF A COMPLEX MATRIX
C                          SUCH THAT ||X I|| = 1.0
C
C************************************************************************
            SUBROUTINE NORM(A,IN)
            IMPLICIT REAL*8(A-H,P-Z)
            COMPLEX*16 A(10,10)
            INTEGER IN
            DO 10 J = 1,IN
                  X = 0.D0
            DO 20 I = 1,IN
20    X = X + DREAL(A(I,J))**2 + DIMAG(A(I,J))**2
            DO 10 I = 1,IN
10    A(I,J) = A(I,J)/DSQRT(X)
            RETURN
            END
```

```
C***********************************************************************
C
C             SUBROUTINE RESULT: WRITES EIGENS RESULTS ON FILE 08
C
C***********************************************************************
            SUBROUTINE RESULT(A,B,C,F,FDEL,BDEL,Z4,Z5,L1,M1,N1)
            IMPLICIT REAL*8(A-H,P-Z)
            REAL*8 A(10,10),B(10,10),C(10,10),F(10,10),FDEL(10,10),BDEL(10,10)
            COMPLEX*16 Z4(10,10),Z5(10),FC(10,10)
            INTEGER L1,M1,N1
            WRITE(8,900)L1,M1,N1
            DO 10 I = 1,N1
10     WRITE(8,1000)(A(I,J),J=1,N1)
            DO 20 I = 1,N1
20     WRITE(8,1000)(B(I,J),J=1,M1)
            DO 30 I = 1,L1
30     WRITE(8,1000)(C(I,J),J=1,N1)
            DO 40 I = 1,M1 .
40     WRITE(8,1000)(F(I,J),J=1,L1)
            DO 50 I = 1,N1
50     WRITE(8,1100)(Z4(I,J),J=1,N1)
            DO 60 I = 1,N1
60     WRITE(8,1000)Z5(I)
            DO 61 I = 1,M1
61     WRITE(8,1000)(FDEL(I,J),J=1,N1)
            DO 62 I = 1,N1
62     WRITE(8,1000)(BDEL(I,J),J=1,M1)
900    FORMAT(3I2)
1000   FORMAT(1X,6E12.5)
1100   FORMAT(1X,6E12.5,/1X,6E12.5)
            RETURN
            END
C**********************************************************************
C
C         SUBROUTINE RANKD: ANALYZES REQUIRED STRUCTURE OF RANK DEFICIENT
C                             MATRICES
C
C***********************************************************************
            SUBROUTINE RANKD(BS,UT,S,N,M,U0TC,U1TC,RZINVC,IRANK,BGI)
            IMPLICIT REAL*8(A-H,P-Z)
            REAL*8 BS(10,10),S(10),SM(10,10),RZ(10,10),RZINV(10,10),
          1 U(10,10),UT(10,10),VT(10,10),SI(10),WK(50),SMI(10,10),WAREA(200)
          2,BGI(10,10),CGI(10,10),BSM(10,10)
            COMPLEX*16 RZINVC(10,10),U0TC(10,10),U1TC(10,10)
            INTEGER N,M,IRANK
            IRANK = M
            DO 1 I = 1,10
            DO 1 J = 1,10
            BGI(I,J) = 0.D0
            RZ(I,J) = 0.D0
            RZINV(I,J) = 0.D0
            U0TC(I,J) = (0.D0,0.D0)
            U1TC(I,J) = (0.D0,0.D0)
            RZINVC(I,J) = (0.D0,0.D0)
            U(I,J) = 0.D0
1      VT(I,J) = 0.D0
            DO 2 I = 1,50
2      WK(I) = 0.D0
            DO 3 I = 1,M
            IF(S(I).GT.0.D0)GOTO 3
            IRANK = I-1
            GOTO 4
3      CONTINUE
4      DO 30 I = 1,N
            DO 30 J = 1,N
            SM(I,J) = 0.D0
30     U(I,J) = UT(J,I)
            DO 40 I = 1,N
            DO 40 J = 1,IRANK
```

```
40    U0TC(J,I) = DCMPLX(U(1,J).0.D0)
            DD 50 I = I,N
            DD 50 J = IRANK + I,N
50    U1TC(J-IRANK,I) = DCMPLX(U(I,J).0.D0)
C     DD 60 I = 1,M
            DD 60 I = I,IRANK
            DO 60 J = 1,M
60    VT(I,J) = BS(J,I)
C     WRITE(6,1000)
C000  FORMAT(/5X,'U MATRIX',/)
C     DO 1020 I = 1,N
C020  WRITE(6,1030)(U(1,J),J=1,N)
C030  FORMAT(IX,6(E12.5,1X))
C     WRITE(6.1040)
C040  FCRMAT(/5X,'S MATRIX',/)
C     DD 1050 I = I,N
C050  WRITE(6,1030) S(I)
C     WRITE(6,1060)
C060  FCRMAT(/5X,'V MATRIX',/)
C     DO 1070 I = 1,M
C070  WRITE(6,I030)(BS(I.J),J=1,M)
            DD 70 I = 1,IRANK
70    SM(I,I) = S(I)
            CALL VMULFF(SM,VT,IRANK,M,M,I0,10,RZ,I0,IER)
C     WRITE(6,2000)
C000  FCRMAT(/5X,'RZ MATRIX',/)
C     DO 2010 I = I,IRANK
C010  WRITE(6,1030)(RZ(I.J),J=1,M)
C     TOL = 0.D0
C     CALL LGINF(RZ,10,IRANK,M,TOL,RZINV,10,SI,WK,IER)
C     WRITE(6,3000)
C000  FDRMAT(/5X,'**** SIGMA MATRIX PRIOR TD INVERSION ****',/)
C     WRITE(6,3030)((SM(I.J),J=I,IRANK),I=1,IRANK)
C     CALL LINV2F(SM,IRANK,10,SMI,0,WAREA,IER)
            DO 3001 I = 1,IRANK
3001  SMI(I,I) = I.D0/SM(I,I)
C     WRITE(6,4000)
C000  FDRMAT(/5X,'**** SIGMA MATRIX INVERTED ****',/)
C     WRITE(6,3030)((SMI(I.J),J=I,IRANK),I=1,IRANK)
C030  FORMAT(IX,3(E12.5,1X))
            CALL VMULFF(BS,SMI,M,IRANK,IRANK,10,10,RZINV,10,IER)
            DD 80 I = I,M
            DD 80 J = 1,IRANK
80    RZINVC(I.J) = DCMPLX(RZINV(I.J).0.D0)
            CALL VMULFF(BS,SMI,M,IRANK,IRANK,10,10,BSM,10,IER)
            CALL VMULFF(BSM,UT,M,IRANK,N,10,10,BGI,I0,IER)
            RETURN
            END
C*****************************************************************
C
C        SUBROUTINE FEEDEF: CALCULATES FEEDBACK GAINS FOR RANK DEFICIENT
C                           B AND/DR C MATRICES VIA FLETCHER, ET.AL.
C                           1985
C*****************************************************************
            SUBROUTINE FEEDEF(A,B,C,V,VDI,EIGDM,BGI,U1TC,CGI,N,M,L,IY,IFEED,FC
          1)
            IMPLICIT REAL*8(A-H,P-Z)
            REAL*8 A(10,10),B(10,10),C(10,10),BGI(10,10),CW(10,I0),
          1 CGI(I0,10),CS0S(10),TOTBS(10),SB(I0),SC(10),WK(200),
          2 CT(I0,10),CTW(10,10),CTUT(10,10),SCT(10),CTGI(10,10),WK1(200),
          3 WA(I0),COBJ,COBJR,COBJL,WEIGHL,WEIGHR,WORK(20)
            COMPLEX*16 AC(10,10),BC(10,10),CC(10,10),V(10,10),FC(10,10),
          1 DQ(I0,10),S0(10,I0),DQN(10,I0),TOT(10,10),TOTA(I0,10),DQT(I0,I0),
          2 U0(10,I0),ACS(10,I0),S0D(I0,I0),BV0(10,I0),BIC(I0,10),CIC(I0,10),
          3 CS0(10,10),TOTB(10,10),CSU(10,10),CSV(10,10),TOTBU(10,10),
          4 TOTBV(10,10),CS0SC(10,10),TOTBSC(10,10),CSUT(10,10),TOTBUT(10,10)
          5,CSVC(10,I0),CS0I(10,10),TOTBSS(10,I0),TOTBI(10,10),WM1(10,10),
          6 WM2(10,10),WM3(10,10),WM4(10,10),WM5(10,10),WM6(10,10),WM7(10,10)
          7,WM8(I0,I0),WM9(10,10),WM10(10,10),WM11(10,10),WM12(10,10),
          8 WM13(10,10),WM14(10,10),Z,EIGDM(10,10),BCS(10,10),CCC(10,10),
          9 T0(10,10),VDI(10,10),BCMP(10,10),CCMP(10,10),CCSS(10,10)
            COMPLEX*16 CSOMP(10,10),WM15(10,10),TTBB(10,10),TOTMP(10,10),
          1 WM16(10,10),WM17(10,10),WM18(10,10),WM20(10,10),WM21(10,10),
          2 WM22(10,10),U1TC(10,10),CT1TC(10,10),CT0TC(10,10),ZINCTC(10,10),
          3 AML(10,10),ATML(10,10),RN(10,10,10),LN(10,10,10),T(10,10),
          4 VW(10,10),WM23(10,10),WM24(10,10),WVR(10),WVL(10),EB(10,10),
          5 NMR(10,10),NML(10,10),RCBJ(10),LOBJ(10),UNITYC(10,10),
          6 UOT(10,10),V0(10,10),FCT(10,10),FCS(10,10),BCFC(10,10),
          7 BCFCC(10,10),CLM(10,10),EVALU(10),EVECT(10,10),V1(10,10),
          8 VE(10,10),VEV(10,10),VMA(10,10),BV(10,10),FCG(10,10),U1C(10,10)
            REAL*4 OBJ,X(50),VLB(50),VUB(50),RA(20,40),G(1),DF(20),WK2(4000)
            INTEGER N,M,L,IFEED,IQ,IN,IRANCT,1Y,IWK(1000),INFO,IDG(1),IC(10)
            WRITE(6,10)
10    FORMAT(/5X,'**** PRESENTLY IN FEEDEF, SINCE RANK(C) < N. ENTER****
          1',/5X,     '**** DESIRED DIMENSION OF R.H. EIGENVECTOR SET FOR****
          2',/5X,     '**** SPLIT EIGENSTRUCTURE COMPUTATION.            ****
          3',/)
            READ(*,15)IQ
15    FORMAT(I1)
            CALL FRTCMS('CLRSCRN ')
            IN = N - IQ
C        INIT MATRICES
            DO 11 I = 1,10
            CSOS(I) = 0.D0
            TOTSS(1) = 0.D0
            SCT(I) = 0.D0
            WA(I) = 0.D0
            WVR(I) = (0.D0,0.D0)
            WVL(I) = (0.D0,0.D0)
            ROBJ(I) = 0.D0
            LQBJ(I) = 0.D0
            DO 11 J = 1,10
            Z = (0.D0,0.D0)
            AC(I,J) = Z
            BC(I,J) = Z
            CC(I,J) = Z
            CW(I,J) = 0.D0
            CT(I,J) = 0.D0
            CTW(I,J) = 0.D0
            CTUT(I,J) = 0.D0
            T(I,J) = Z
            VW(1,J) = Z
            WM23(I,J) = Z
            WM24(I,J) = Z
            NMR(I,J) = Z
            NML(1,J) = Z
            DQ(I,J) = Z
            S0(1,J) = Z
            DQN(I,J) = Z
            TOT(I,J) = Z
            TOTA(I,J) =Z
            DQT(I,J) = Z
            U0(I,J) =  Z
            ACS(I,J) = Z
            SOD(I,J) = Z
            BV0(I,J) = Z
            B1C(I,J) = Z
            UOT(I,J) = Z
            V0(I,J) = Z
            CIC(I,J) = Z
            CS0(I,J) = Z
            TOTB(I,J) =Z
            CSU(I,J) = Z
            CSV(I,J) = Z
            TOTBU(I,J) =Z
            TOTBV(I,J) =Z
            CSOSC(I,J) =Z
            TOTBSC(1,J) = Z
          CSUT(I,J) =   Z
          TOTBUT(I,J) = Z
          CSVC(I,J) = Z
          UNITYC(I,J) = (0.D0,0.D0)
11    CONTINUE
C        BUILD CONSTANTS
          INFO = 0
          WEIGHR = 1.D0
          WEIGHL = 1.D0
          DO 12 I = I,N
          UNITYC(I,I) = (1.D0,0.D0)
          DO 12 J = 1,N
          T(I,J) = VDI(J,I)
12    AC(I,J) = DCMPLX(A(I,J),0.D0)
          DO 13 I = 1,N
          DO 13 J = 1,M
13    BC(I,J) = DCMPLX(B(I,J),0.D0)
          DO 14 I = I,L
          DO 14 J = 1,N
          CW(I,J) = C(I,J)
14    CC(I,J) = DCMPLX(C(I,J),0.D0)
C        CHECK IF LEFT EIGENVECTORS ARE MEMBERS OF N(C1**T*(A-LAM*I))
C             FIND SVD OF C**T
          DO 200 I = I,L
          DO 200 J = I,N
          CTUT(J,J) = 1.D0
          CTW(J,I) = C(I,J)
200   CT(J,I) = C(I,J)
          CALL LSVDF(CTW,10,N,L,CTUT,10,N,SCT,WK1,IER)
          CALL RANKD(CTW,CTUT,SCT,N,L,CT0TC,CT1TC,ZINCTC,IRANCT,CTGI)
C
C        BUILD NULL SPACE OPERATORS FOR B AND C**T
C
          DO 210 I = 1,N
          DO 205 J = 1,N
          DO 205 K = 1,N
          AML(J,K) = AC(J,K) - EIGDM(I,I)*UNITYC(J,K)
205   ATML(J,K) = AC(K,J) - EIGDM(I,I)*UNITYC(J,K)
          CALL CMATML(U1TC,AML,IY,N,N,WM23)
          CALL CMATML(CT1TC,ATML,N-IRANCT,N,N,WM24)
          DO 206 JJ = 1,IY
          DO 206 KK = I,N
          U1C(KK,JJ) = U1TC(JJ,KK)
206   RN(I,JJ,KK) = WM23(JJ,KK)
          DO 207 JJ = 1,N-IRANCT
          DO 207 KK = 1,N
207   LN(I,JJ,KK) = WM24(JJ,KK)
210   CONTINUE
C
C        FIND OBJECTIVE FUNCTION
C
208   COBJ = 0.D0
          COBJR = 0.D0
          COBJL = 0.D0
          ICONJG = 0
          DO 220 I = 1,N
          IF(ICONJG.EQ.1)GOTO 219
          DO 225 K = 1,N
          Y = DIMAG(EIGDM(I,I))
          IF(Y.EQ.0.D0)GOTO 209
          IF(K.NE.N)GOTO 209
          ICONJG = 1
209   WVR(K) = V(K,I)
225   WVL(K) = T(K,I)
          DO 226 JJ = 1,IY
          DO 226 KK = 1,N
226   NMR(JJ,KK) = RN(I,JJ,KK)
          DO 227 JJ = 1,N-IRANCT
          DO 227 KK = I,N
```

```
227   NML(JJ,KK) = LN(I,JJ,KK)
         CALL CMATML(NMR,WVR,IY,N,1,ROBJ)
         CALL CMATML(NML,WVL,N-IRANCT,N,1,LOBJ)
         DO 300 JJ = 1,IY
300   COBJR = COBJR + CDABS(ROBJ(JJ))
         DO 301 JJ = 1,N-IRANCT
301   COBJL = COBJL + CDABS(LOBJ(JJ))
         GOTO 220
219   ICONJG = 0
220   CONTINUE
         COBJ = WEIGHR*COBJR + WEIGHL*COBJL
         OBJ = SNGL(COBJ)
         IF(INFO.EQ.1)GOTO 241
         WRITE(6,230)COBJR,COBJL
230   FORMAT(/5X,'**** SUBSPACE COMPATIBILILTY TEST PERFORMED. ****',
     1       /5X,'**** RT SUBSPACE RESIDUAL = ',E12.5,'  ****',
     2       /5X,'**** LT SUBSPACE RESIDUAL = ',E12.5, '    ****',
     3       /5X,'**** ENTER "1" FOR ADS MINIMIZATION             ****',
     4       /5X,'**** ENTER "2" IF SUBSPACE RESIDUAL OK.         ****',
     5       /5X,'**** ENTER "3" IF YOU DESIRE TO USE PREVIOUSLY ****',
     6       /5X,'**** OPTIMIZED RT AND LEFT EIGENVECTOR SETS.    ****',/
     7)
         READ(*,231)IMIN
231   FORMAT(I1)
         CALL FRTCMS('CLRSCRN ')
         IF(IMIN.EQ.3)GOTO 510
         IF(IMIN.EQ.2)GOTO 16
         WRITE(6,260)
260   FORMAT(/5X,'** ENTER WEIGHT FACTOR FOR RIGHT EIGENSTRUCTURE **',/)
         READ(*,237)WEIGHR
         WRITE(6,261)
261   FORMAT(/5X,'** ENTER WEIGHT FACTOR FOR LEFT EIGENSTRUCTURE **',/)
         READ(*,237)WEIGHL
         CALL FRTCMS('CLRSCRN ')
C*************************************************************************
C
C           OPTIMIZER CODE FOR LEFT AND RIGHT SUBSPACE RQMTS
C           ONLY BUILD ONE DESIGN VARIABLE VECTOR AND UPPER/
C           LOWER BOUND VECTOR PER COMPLEX EIGENVALUE. AFTER
C           OPTIMIZATION, INVOKE CONJUGATE VECTOR RQMT.
C
C*************************************************************************
232   ISTRAT = 0
         ICPT = 3
         IONED = 1
         NOV = 2*N*N
         NCON = 0
         IGRAD = 0
         IPRINT = 1000
         NGT = 10
         NRA = 10
         NCOLA = 10
C        SET BOUNDS
         WRITE(6,235)
235   FORMAT(/5X,'**** AN ACCEPTABLE SET OF R.H. EIGENVECTORS ****',/5X,
     1            '**** HAVE BEEN COMPUTED IN KVECT. ENTER ABS ****',/5X,
     2            '**** VALUE OF ELEMENT TOL. TO SATISFY L.H.  ****',/5X,
     3            '**** SUBSPACE REQUIREMENT.                  ****',/)
         WRITE(6,236)
236   FORMAT(/5X,'**** REAL TOLERANCE =  ****',/)
         READ(*,237)BR
         WRITE(6,238)
238   FORMAT(/5X,'**** IMAG TOLERANCE =  ****',/)
         READ(*,237)BI
237   FORMAT(F12.5)
         CALL FRTCMS('CLRSCRN ')
C        BUILD DESIGN VARIABLE VECTOR AND UPPER/LOWER BOUND VECTOR
239   JJ = 1
         ICONJG = 0
```

```
          DO 240 KK = 1.N
          IF(ICONJG.EQ.1)GOTO 242
          DO 240 LL = 1.N
          RX = DREAL(V(LL.KK))
          CX = DIMAG(V(LL.KK))
          Y = DIMAG(EIGDM(KK.KK))
          IF(Y.EQ.0.D0)GOTO 280
281    IF(LL.NE.N)GOTO 280
262    NDV = NDV - 2*N
          ICONJG = 1
280    X(JJ) = SNGL(RX)
          X(JJ+1) = SNGL(CX)
          VUB(JJ) = X(JJ) + BR
          VLB(JJ) = X(JJ) - BR
          IF(CX.NE.0.D0)GOTO 286
          VUB(JJ+1) = 0.0
          VLB(JJ+1) = 0.0
          GOTO 285
286    VUB(JJ+1) = X(JJ+1) + BI
          VLB(JJ+1) = X(JJ+1) - BI
285    IC(JJ) = 0
          IC(JJ+1) = 0
          DF(JJ) = 0.
          DF(JJ+1) = 0.
          GOTO 265
242    ICONJG = 0
          GOTO 240
265    JJ = JJ + 2
240    CONTINUE
241    CALL ADS(INFO.ISTRAT.IOPT.IONED.IPRINT.IGRAD.NDV.NCON.X.VLB.VUB.
        1 OBJ.G.IDG.NGT.IC.DF.RA.NRA.NCOLA.WK2.4000.IWK.1000)
C         EVALUATE OBJECTIVE FUNCTION
          JJ = 1
          ICONJG = 0
          DO 245 KK = 1.N
          IF(ICONJG.EQ.1)GOTO 243
          DO 245 LL = 1.N
          IF(DIMAG(EIGDM(KK.KK)).EQ.0.D0)GOTO 246
270    IF(LL.NE.N)GOTO 246
271    ICONJG = 1
246    V(LL.KK) = CMPLX(X(JJ).X(JJ+1))
          VW(LL.KK) = V(LL.KK)
          GOTO 248
243    DO 244 LLL = 1.N
          V(LLL.KK) = DCONJG(V(LLL.KK-1))
244    VW(LLL.KK) = V(LLL.KK)
          ICONJG = 0
          GOTO 245
248    JJ = JJ + 2
245    CONTINUE
          CALL LEQT1C(VW.N.10.UNITYC.N.10.0.WA.IER)
          DO 247 KK = 1.N
          DO 247 LL = 1.N
247    T(LL.KK) = UNITYC(KK.LL)
          DO 250 KK = 1.N
          DO 250 LL = 1.N
250    UNITYC(KK.LL) = (0.D0.0.D0)
          DO 252 KK = 1.N
252    UNITYC(KK.KK) = (1.D0.0.D0)
          IF(INFO.EQ.0)GOTO 16
          GOTO 208
C         BUILD S0 AND TOT.DO AND DON
16     WRITE(6.251)COBJR.COBJL
251    FORMAT(/5X.'**** RT SUBSPACE RESIDUAL  = '.E12.5.'  ****'./5X.
        1          '**** LT SUBSRACE RESIDUAL  = '.E12.5.'  ****'./5X.
        2          '**** ENTER "1" TO RETURN TO ADS. ELSE    ****'./5X.
        3          '**** ENTER "2".                          ****'./)
          READ(*.231)IRTN
          CALL FRTCMS('CLRSCRN ')
          INFO = 0
          IF(IRTN.EQ.1)GOTO 20B
          CALL COND(V,N,VCOND)
          CALL COND(T,N,TCOND)
C        DEBUG WRITE
          WRITE(6,290)
290   FORMAT(/5X,'**** OPTIMIZED RT. EIGENVECTOR MATRIX ****',/)
          WRITE(6,291)((V(I,J),J=1,N),I=1,N)
291   FORMAT(2X,B(E9.1,1X))
          WRITE(6,520)VCOND,COBJR
520   FORMAT(/5X,'**** RT. EV. 2 NORM COND. NO. = ',E12.5,' ****',/5X,
     1            '**** RT. SUBSPACE RESIDUAL = ',E12.5,' ****',/)
          WRITE(6,292)
292   FORMAT(/5X,'**** OPTIMIZED LT. EIGENVECTOR MATRIX ****',/)
          WRITE(6,291)((T(I,J),J=1,N),I=1,N)
          WRITE(6,521)TCOND,COBJL
521   FORMAT(/5X,'**** LT. EV. 2 NORM COND. NO. = ',E12.5,' ****',/5X,
     1            '**** LT. SUBSPACE RESIDUAL = ',E12.5,' ****',/)
C        SAVE RT/LT EIGENVECTORS,COND NOS. AND RESIDUALS ON FILEDEF 10
          WRITE(10,500)((V(I,J),J=1,N),I=1,N)
          WRITE(10,501)VCOND,COBJR
          WRITE(10,500)((T(I,J),J=1,N),I=1,N)
          WRITE(10,501)TCOND,COBJL
501   FORMAT(1X,E12.5,1X,E12.5)
          GOTO 511
500   FORMAT(1X,4(E12.5,1X))
510   READ(10,500)((V(I,J),J=1,N),I=1,N)
          READ(10,501)VCOND,COBJR
          READ(10,500)((T(I,J),J=1,N),I=1,N)
          READ(10,501)TCOND,COBJL
          WRITE(6,520)VCOND,COBJR
          WRITE(6,521)TCOND,COBJL
511   DO 20 I = 1,N
          DO 20 J = 1,IQ
          DO(J,J) = EIGDM(J,J)
20    SO(I,J) = V(I,J)
          DO 25 I = 1,N
          DO 25 J = N-IQ+1,N
          DON(J-N+IQ,J-N+IQ) = EIGDM(J,J)
25    TOT(J-N+IQ,I) = T(I,J)
C        DEBUG WRITES AND CONDITION CHECKS
          CALL CMATML(TOT,SO,IQ,N,IQ,WM22)
C     WRITE(6,41)
C1    FORMAT(/5X,'**** TOT * SO ****',/)
C     WRITE(6,42)((WM22(I,J),J=1,IQ),I=1,IQ)
C2    FORMAT(1X,4(1X,E12.5,1X))
C     WRITE(6,26)
C6    FORMAT(/5X,'**** SO FOLLOWED BY TO ****',/)
C     WRITE(6,27)((DREAL(SO(I,J)),J=1,IQ),I=1,N)
C7    FORMAT(2X,2(1X,E12.5,2X))
C     WRITE(6,27)((DREAL(TOT(I,J)),I=1,IQ),J=1,N)
C     WRITE(6,28)
C8    FORMAT(/5X,'**** DO FOLLOWED BY DON ****',/)
C     WRITE(6,27)((DREAL(DO(I,J)),J=1,IQ),I=1,IQ)
C     WRITE(6,27)((DREAL(DON(I,J)),J=1,IN),I=1,IN)
C        BUILD U0,BV0
          CALL CMATML(TOT,AC,IQ,N,N,TOTA)
          CALL CMATML(DON,TOT,IQ,IQ,N,DOT)
          DO 30 I = 1,IQ
          DO 30 J = 1,N
30    U0(I,J) = TOTA(I,J) - DOT(I,J)
          CALL CMATML(AC,SO,N,N,IQ,ACS)
          CALL CMATML(SO,DO,N,IQ,IQ,SOD)
          DO 35 I = 1,N
          DO 35 J = 1,IQ
35    BV0(I,J) = ACS(I,J) - SOD(I,J)
C         DEBUG WRITES
C     WRITE(6,36)
C6    FORMAT(/5X,'**** BV0 ****',/)
```

```
C     WRITE(6.37)((DREAL(BV0(I.J)).J=1.IQ).I=1.N)
C7    FORMAT(1X.2(1X.E12.5.1X))
C     WRITE(6.38)
C8    FORMAT(/5X.'**** UOTC ****'./)
C     WRITE(6.39)((DREAL(U0(I.J)).J=1.N).I=I.IN)
C9    FORMAT(1X.4(1X.E12.5.1X))
C        FIND BINV.CINV
C     CALL LGINF(BW.10.N.M.0.D0.BGI.10.SB.WK.IER)
C     CALL LGINF(CW.10.L.N.0.D0.CGI.10.SC.WK.IER)
           DO 40 I = 1.M
           DO 40 J = 1.N
40    BIC(I.J) = DCMPLX(BGI(I.J).0.D0)
           DO 50 I = 1.N
           DO 50 J = 1.L
50    CIC(I.J) = DCMPLX(CGI(I.J).0.D0)
C        CALCULATE U0**T AND V0
           CALL CMATML(U0.CIC.IQ.N.L.U0T)
           CALL CMATML(BIC.BV0.M.N.IQ.V0)
C        DEBUG WRITES
           CALL CMATML(BC.BIC.N.M.N.BCB)
           CALL CMATML(CC.CIC.L.N.L.CCC)
           CALL CMATML(BCB.BC.N.N.M.BCMP)
           CALL CMATML(CCC.CC.L.L.N.CCMP)
           CALL CMATML(BIC.BC.M.N.M.WM20)
           CALL CMATML(CC.CIC.L.N.L.WM21)
C     WRITE(6.51)
C1    FORMAT(/5X.'**** MOORE PENROSE B*PINV(B)*B RESULTS ****'./)
C     WRITE(6.52)((DREAL(BCMP(I.J)).J=1.M).I=1.N)
C2    FORMAT(1X.2(1X.E12.5.1X))
C     WRITE(6.53)
C3    FORMAT(/5X.'**** MOORE PENROSE C*PINV(C)*C RESULTS ****'./)
C     WRITE(6.54)((DREAL(CCMP(I.J)).J=1.N).I=I.L)
C4    FORMAT(1X.4(1X.E12.5.1X))
C     WRITE(6.55)
C5    FORMAT(/5X.'**** B+ * B ****'./)
C     WRITE(6.52)((DREAL(WM20(I.J)).J=I.M).I=I.M)
C     WRITE(6.56)
C6    FORMAT(/5X.'**** C * C+ ****'./)
C     WRITE(6.57)((DREAL(WM21(I.J)).J=1.L).I=1.L)
C7    FORMAT(1X.3(1X.E12.5.1X))
C        BUILD CS0+ AND (TOTB)+
           CALL CMATML(CC.S0.L.N.IQ.CS0)
           CALL CMATML(CC.S0.L.N.IQ.WM17)
           CALL CMATML(TOT.BC.IQ.N.M.TOTB)
           CALL CMATML(TOT.BC.IQ.N.M.WM18)
           CALL CSVD(CS0.10.10.L.IQ.0.L.IQ.CS0S.CSU.CSV)
           CALL CSVD(TOTB.10.10.IQ.M.0.IQ.M.TOTBS.TOTBU.TOTBV)
           IRANKC = IQ
           DO 60 I = 1.IQ
           IF(CS0S(I).GT.0.D0)GOTO 60
           IRANKC = I - I
           GOTO 61
60    CONTINUE
61    IRANKT = M
           DO 65 I = 1.M
           IF(TOTBS(I).GT.0.D0)GOTO 65
           IRANKT = I - 1
           GOTO 62
65    CONTINUE
62    DO 70 I = 1.IRANKC
           XV = 1.D0/CS0S(I)
70    CS0SC(I.I) = DCMPLX(XV.0.D0)
           DO 75 I = 1.IRANKT
           XV = 1.D0/TOTBS(I)
75    TOTBSC(I.I) = DCMPLX(XV.0.D0)
           DO 76 I = 1.L
           DO 76 J = I.L
76    CSUT(J.I) = CSU(I.J)
           DO 77 I = 1.IQ
```

```
            DO 77 J = 1,IQ
77    TOTBUT(J,I) = TOTBU(I,J)
            CALL CMATML(CSV,CSOSC,IQ,IRANKC,IRANKC,CSVC)
            CALL CMATML(CSVC,CSUT,IQ,IRANKC,L,CSOI)
            CALL CMATML(TOTBV,TOTBSC,M,IRANKT,IRANKT,TOTBSS)
            CALL CMATML(TOTBSS,TOTBUT,M,IRANKT,IQ,TOTBI)
C        MOORE PENROSE TEST FOR CSOI AND TOTBI
            CALL CMATML(WMI7,CSOI,L,IQ,L,CCSS)
            CALL CMATML(CCSS,WM17,L,L,IQ,CSOMP)
            DO 91 I = I,L
            DO 91 J = I,IQ
91    WM15(I,J) = CSOMP(I,J) - WM17(I,J)
C        DEBUG WRITES
C     WRITE(6,90)
C0    FORMAT(/5X,'**** CSO ****
C    I',/)
C     WRITE(6,52)((DREAL(WM17(I,J)),J=1,IQ),I=1,L)
            CALL CMATML(WM18,TOTBI,IN,M,IN,TTBS)
            CALL CMATML(TTB8,WM18,IN,IN,M,TOTMP)
            DO 92 I = 1,IN
            DO 92 J = 1,M
92    WM16(I,J) = TOTMP(I,J) - WM18(I,J)
C        DEBUG WRITES
C     WRITE(6,93)
C3    FCRMAT(/5X,'**** TOTB ****'
C    1,/)
C     WRITE(6,52)((DREAL(WM18(I,J)),J=1,M),I=1,IN)
C        FIND F1T
            CALL CMATML(WM20,TOTBI,M,M,IQ,WM1)
            CALL CMATML(WM1,UOT,M,IQ,L,WM3)
            CALL CMATML(WM3,CCC,M,L,L,WM5)
C        FIND F2T
            CALL CMATML(WM20,VO,M,M,IQ,WM6)
            CALL CMATML(WM6,CSOI,M,IQ,L,WM7)
            CALL CMATML(WM7,CCC,M,L,L,WM8)
C        FIND F3T
            CALL CMATML(WM20,TOTBI,M,M,IQ,WM9)
            CALL CMATML(WM9,WM18,M,IQ,M,WM11)
            CALL CMATML(WM11,VO,M,M,IQ,WM12)
            CALL CMATML(WM12,CSOI,M,IQ,L,WMI3)
            CALL CMATML(WM13,CCC,M,L,L,WMI4)
C        FIND FCT
            DO 85 I = I,M
            DO 85 J = I,L
85    FCT(I,J) = -WM5(I,J) - WM8(I,J) + WM14(I,J)
C        FIND F1S
            CALL CMATML(WM20,VO,M,M,IQ,WM6)
            CALL CMATML(WM6,CSOI,M,IQ,L,WM7)
            CALL CMATML(WM7,CCC,M,L,L,WM8)
C        FIND F2S
            CALL CMATML(WM20,TOTBI,M,M,IQ,WM9)
            CALL CMATML(WM9,UOT,M,IQ,L,WMI0)
            CALL CMATML(WM10,CCC,M,L,L,WM11)
C        FIND F3S
            CALL CMATML(WM20,TOTBI,M,M,IQ,WM12)
            CALL CMATML(WMI2,UOT,M,IQ,L,WM13)
            CALL CMATML(WM13,WM17,M,L,IQ,WM14)
            CALL CMATML(WMI4,CSOI,M,IQ,L,WM15)
            CALL CMATML(WMI5,CCC,M,L,L,WM16)
C        FIND FCS
            DO 80 I = I,M
            DO 80 J = 1,L
80    FCS(I,J) =-WM8(I,J) - WM11(I,J) + WM16(I,J)
C        FIND FCG
            DO 350 I = 1,N
            DO 350 J = I,N
350   VI(I,J) = T(J,I)
            CALL CMATML(V,EIGDM,N,N,N,VE)
            CALL CMATML(VE,VI,N,N,N,VEV)
```

```
          DO 355 I = 1.N
          DO 355 J = 1.N
355   VMA(I.J) = VEV(I.J) - AC(I.J)
          CALL CMATML(U1TC.VMA.IY.N.N.EB)
          WRITE(6.356)
356   FORMAT(/5X.'**** EB ****',/)
          WRITE(6.315)((EB(I.J).J=1.N).I=1.IY)
          CALL CMATML(BIC.VMA.M.N.N.BV)
          CALL CMATML(BV.CIC.M.N.L.FCG)
C         DEBUG WRITE BOTH FCS AND FCT
          WRITE(6.310)
310   FORMAT(/5X.'**** RIGHT FEEDBACK GAINS ****',/)
          WRITE(6.315)((FCS(I.J).J=1.L).I=1.M)
          WRITE(6.311)
311   FORMAT(/5X.'**** LEFT FEEDBACK GAINS ****',/)
          WRITE(6.315)((FCT(I.J).J=1.L).I=1.M)
315   FORMAT(1X.4(1X.E12.5.1X))
          WRITE(6.316)
316   FORMAT(/5X.'**** GAV FEEDBACK GAINS ****',/)
          WRITE(6.315)((FCG(I.J).J=1.L).I=1.M)
          WRITE(6.312)
312   FORMAT(/5X.'**** SELECT RIGHT OR LEFT GAINS ****'./5X.
     1            '**** ENTER "1" FOR RIGHT GAINS  ****'./5X.
     2            '**** ENTER "2" FOR LEFT GAINS   ****'./5X.
     3            '**** ENTER "3" FOR GAV GAINS    ****',/)
          READ(*.231)IG
          IF(IG.EQ.1)GOTO 320
          IF(IG.EQ.3)GOTO 325
          DO 313 I = 1.M
          DO 313 J = 1.L
313   FC(I.J) = FCT(I.J)
          GOTO 1000
320   DO 314 I = 1.M
          DO 314 J = I.L
314   FC(I.J) = FCS(I.J)
          GOTO 1000
325   DO 326 I = I.M
          DO 326 J = I.L
326   FC(I.J) = FCG(I.J)
C        FIND EIGENSTRUCTURE
1000  CALL CMATML(BC.FC.N.M.L.BCFC)
          CALL CMATML(SCFC.CC.N.L.N.BCFCC)
          DO 600 I = 1.N
          DO 600 J = 1.N
600   CLM(I.J) = AC(I.J) + BCFCC(I.J)
          CALL EIGCC(CLM.N.10.1.EVALU.EVECT.10.WORK.IER)
          WRITE(6.610)
610   FORMAT(/5X.'**** EIGENVALUES RESULTING FROM FEEDEF CALC ****',/)
          WRITE(6.620)(EVALU(I).I=1.N)
620   FORMAT(1X.E12.5.1X.E12.5)
          WRITE(6.630)
630   FORMAT(/5X.'**** EIGENVECTORS RESULTING FROM FEEDEF CALC ****',/)
          WRITE(6.291)((EVECT(I.J).J=1.N).I=1.N)
          RETURN
          END
C***********************************************************************
C
C              SUBROUTINE ALGSLN : SOLVES FOR BASE EIGENVECTORS
C                           ALGEGRAICALLY
C
C***********************************************************************
          SUBROUTINE ALGSLN(A.B.C.EIGD.E.UNITY.N1.M1.L1.V.WMAT)
          IMPLICIT REAL*8(A-H.P-Z)
          REAL*8 A(10.10).B(10.3).C(10.10).UNITY(10.10).
        1 AWK1(250).AWK2(250).ALSLAM(10.10).ALINV(10.10).
        2 W2(10.10).W3(10.10).W4(10.10).W5(10.10).W6(10.10).W7(10).W8(10).
        3 WMAT(10.10).W9(10).S(20).WK(30)
          COMPLEX *16 EIGD(10).E(10.10).V(10.10)
          INTEGER N1.M1.L1.III
```

```
C        CALCULATE SLAMDHA
            DO 1000 I = 1,NI
C        REINIT SLAMDHA
            DO 10 II = I,10
            DO 10 JJ = I,10
            ALSLAM(11,JJ) = 0.D0
10    CONTINUE
C        RECOMPUTE ALSLAM FOR EACH EIGENVALUE
            DO 30 J = 1,NI
            DO 30 K = 1,NI
            ALSLAM(J.K) = DREAL(EIGD(I))*UNITY(J.K) - A(J.K)
30    CONTINUE
C        WRITE FOR DEBUG
C     DO 35 J = 1,N1
C     WRITE(6,36)(ALSLAM(J,K),K=1,NI)
C6    FORMAT(1X,6(E10.3,1X))
C5    CONTINUE
C        COMPUTE INVERSE OF ALSLAM
            III = 1
            CALL LINV2F(ALSLAM,N1,10,ALINV,I,AWK1,IER)
C     WRITE(6,100) III
C        COMPUTE C * INV<ALSLAM>
            CALL VMULFF(C.ALINV.L1.NI.N1.10,10.W2.10,IER)
            III = III + 1
C     WRITE(6,100)III
C        COMPUTE C * INV<ALSLAM> * B
            CALL VMULFF(W2,B,L1,N1,M1,10,10,W3,10,IER)
            III= III + 1
C     WRITE(6,100)III
C        COMPUTE INV<C* INV<ALSLAM> * B>
C        DETERMINE IF PSEUDO INVERSE IS REQUIRED
            IF(L1.EQ.M1)GOTO 70
C     CALL PINV(W3,LI,M1,W4)
            CALL LGINF(W3,I0,L1,M1,0.D0,W4,10,S,WK,IER)
            GOTO 80
70    CALL LINV2F(W3,L1,10,W4,I,AWK2,IER)
80    III = III + 1
C     WRITE(6,100)III
C        COMPUTE B * INV<C* INV<ALSLAM> * B>
            CALL VMULFF(B,W4,N1,M1,M1,I0,10,W5,10,IER)
            III = III + 1
C     WRITE(6,100)III
C        COMPUTE ALINV * <B*INV<C*INV<ALSLAM>*B>>
            CALL VMULFF(ALINV,W5,N1,N1,M1,10,10,W6,10,IER)
            III = III + 1
C     WRITE(6,100)III
C00   FORMAT(/5X,'OP NO.',I2,' DONE IN ALGSLN')
C        COMPUTE W6 * E ANO THE W MATRIX
            DO 40 J = 1,L1
            W7(J) = DREAL(E(J,I))
40    CONTINUE
            CALL VMULFF(W6,W7,N1,MI,1,10,I0,W8,I0,IER)
            CALL VMULFF(W4,W7,M1,L1,1,I0,10,W9,10,IER)
            DO 50 J = 1,N1
            V(J,I) = DCMPLX(W8(J),0.D0)
50    CONTINUE
            DO 60 J = 1,MI
            WMAT(J,I) =-W9(J)
60    CONTINUE
I000  CONTINUE
            RETURN
            END
C************************************************************************
C
C             SUBROUTINE PINV: COMPUTES THE PSEUDO INVERSE OF AN
C                              IRXIC REAL MATRIX. OUTPUT IS THE
C                              ICXIR PSEUDO INVERSE (SEE GOLUB AND
C                              VAN LOAN, "MATRIX COMPUTATIONS".P. 139
C     CAUTION::: IF WARNING FLAG RESULTS DUE TO ILL CONDITIONED A**T * A
```

```
C                 RECOMMEND USING IMSL "LGINF" VICE PINV
C
C*************************************************************************
C     SUBROUTINE PINV(A.IR.IC.PIA)
C     IMPLICIT REAL*8(A-H.P-Z)
C     REAL*8 A(10.10).PIA(10.10).AT(10.10).WM(10.10).WAREA(250).
C    1 WM1(10.10).PA(10.10).APA(10.10).T(10.10)
C     INTEGER IR.IC
C        FIND A TRANSPOSE
C     DO 10 I = 1.IR
C     DO 10 J = 1.IC
C     AT(J.I) = A(I.J)
C0    CONTINUE
C        FIND AT * A
C     CALL VMULFF(AT.A.IC.IR.IC.10.10.WM.10.IER)
C          WRITE WM FOR PROGRAM DEBUG
C     DO 2001II = 1.IC
C     WRITE(6.2000)(WM(II.K).K=1.IC)
C000  FORMAT(1X.6(E10.3.1X))
C001  CONTINUE
C        FIND INV<AT * A>
C     CALL LINV2F(WM.IC.10.WM1.1.WAREA.IER)
C        FIND INV<AT * A> * AT
C     CALL VMULFF(WM1.AT.IC.IC.IR.10.10.PIA.10.IER)
C        DETERMINE IF PSEUDO-INVERSE SATISFIES THE MOORE-PENROSE
C        CONDITION:     AXA = A
C     CALL VMULFF(PIA.A.IC.IR.IC.10.10.PA.10.IER)
C     CALL VMULFF(A.PA.IR.IC.IC.10.10.APA.10.IER)
C     DO 15 I = 1.IR
C     DO 15 J = 1.IC
C     T(I.J) = DABS(A(I.J) - APA(I.J))
C5    CONTINUE
C     WRITE(6.30)
C0    FORMAT(//5X.'**** MOORE-PENROSE TEST FOR PSEUDO-INV ****'./5X.
C    1             '****   ABSOLUTE VALUE(A - AXA)        ****'./)
C     DO 50 I = 1.IR
C     WRITE(6.40)(T(I.J).J=1.IC)
C0    FORMAT(1X.6(E10.3.1X))
C0    CONTINUE
C     RETURN
C     END
```

APPENDIX D

RECONF

```
C------------------------------------------------------------------------
C
C             RECONF: RECONFIGURATION ALGORITHM FOR THE F18
C                     DYNAMIC MODEL
C             FILE 01 = RECONF DATA = UNDAMAGED F-18 SYSTEM MATRICES
C                       (READ) ----OR----
C                       RECOXX DATA = DAMAGED F-18 SYSTEM MATRICES
C                       (READ)
C             FILE 02 = UNDEIG DATA = UNDAMAGED F18 EIGENSTRUCTURE
C                       (WRITE) OR (READ) ----OR----
C                       OPTEIG DATA = RECONFIGURED F18 EIGENSTRUCTURE
C                       (READ)
C             FILE 03 = MATDREC DATA FOR "CONTROLS" (OPTMATD)
C                       (WRITE)
C             FILE 04 = PLOTREC DATA FOR "CONTROLS" (OPTPLOT)
C                       (WRITE)
C             FILE 07 = FRECCN (RECONFIGURED GAINS)
C                       (WRITE) OR (READ)
C             FILE 08 = UNDABC DATA = UNDAMAGED F18 MATRICES FOR NULL
C                       SPACE ANALYSIS FOR ELASPACE (WRITE SNGL PREC)
C             FILE 09 = ERSPACE (SLOW RIGHT EIGENSTRUCTURE)
C                       (WRITE)
C             FILE 10 = RECEIG DATA = RECONFIGURED EIGENSTRUCTURE
C                       (WRITE)
C             FILE 12 = FXACT DATA = U81TC
C                       (WRITE DURING K = 5)
C------------------------------------------------------------------------
C
C                       CDR V.F. GAVITO
C                           JUNE 1986
C
C------------------------------------------------------------------------
            IMPLICIT REAL*8(A-H,O-Z)
            REAL*4 ASING(55,55),BSING(55,10),CSING(18,55)
            DIMENSION A(55,55),B(55,10),C(18,55),G1(55,3),G2(10,3),F(10,18)
            DIMENSION FTILDA(10,18),CLM(55,55),BS(55,55),U8T(55,55),S8(55)
            DIMENSION WK8(110),CS(55,55),UCT(55,55),SC(55),WKC(110)
            DIMENSION FTC(10,55),BF(55,55),CLMTIL(55,55),WKT(200)
            DIMENSION WK(200),UNITY(55,55),WK1(3200),WA1(55)
            DIMENSION AF18(55,55),BF18(55,3),CF18(18,55),DF18(55,3)
            DIMENSION NAF18(2),NBF18(2),NCF18(2),NDF18(2)
            DIMENSION U(3),XX(55),DUM4(55),DUM5(55)
            DIMENSION NU(2),NXX(2),NDUM4(2),NDUM5(2),BGI(55,55),CGI(55,55)
            DIMENSION CT(55,55)
            COMPLEX*16 E(55),X(100,100),DEL(55,55),UNITYC(55,55),XDX(55,55),
          1 XD(55,55),XI(55,55),XS(55,55),XDXA(55,55)
            COMPLEX*16 UB0TC(55,55),U81TC(55,55),BZINV(55,55)
            COMPLEX*16 UC0TC(55,55),UC1TC(55,55),CZINV(55,55)
            COMPLEX*16 BUB(55,55),CUC(55,55),BUXD(55,55),FC(55,55)
            COMPLEX*16 ETIL(55),XTIL(100,100),BGIC(55,55),CGIC(55,55)
            COMPLEX*16 ESLOW(24),XSLOW(55,24),ES(55),SR,O(55,55)
            COMPLEX*8 ESING(24)
            INTEGER K,IMOD,IWRITE,IEVECT,IELE,ISLOW,KFAST,IROW,IDIS,IRESP,
          *IRANKB,IRANKC,IROWS,ISTATE(55),KSTATE(55),IORDER,IFULL,NCONT,
          *NST,NSTP,IDAM
            CHARACTER*4 NAME
C
C-----------------------INITIALIZE MATRICES-----------------------------
C
            DO 10 I = 1,55
            ES(I) = 0.D0
            ISTATE(I) = 0
            DO 10 J = 1,55
            UNITY(I,J) = 0.D0
            UNITYC(I,J) = DCMPLX(0.D0,0.D0)
            CLM(I,J) = 0.D0
            DEL(I,J) = DCMPLX(0.D0,0.D0)
            XDX(I,J) = DCMPLX(0.D0,0.D0)
            XD(I,J) = DCMPLX(0.D0,0.D0)
            XI(I,J) = DCMPLX(0.D0,0.D0)
            X(I,J) = DCMPLX(0.D0,0.D0)
            BUXD(I,J) = DCMPLX(0.D0,0.D0)
            UST(I,J) = 0.D0
            UCT(I,J) = 0.D0
            BGI(I,J) = 0.D0
            CGI(I,J) = 0.D0
            US0TC(I,J) = DCMPLX(0.D0,0.D0)
            UB1TC(I,J) = DCMPLX(0.D0,0.D0)
            BZINV(I,J) = DCMPLX(0.D0,0.D0)
            CZINV(I,J) = DCMPLX(0.D0,0.D0)
            BS(I,J) = 0.D0
            CS(I,J) = 0.D0
            BGIC(I,J) = DCMPLX(0.D0,0.D0)
            CGIC(I,J) = DCMPLX(0.D0,0.D0)
  10   A(I,J) = 0.D0
            DO 15 I = 1,55
            UNITY(I,I) = 1.D0
            UNITYC(I,I) = DCMPLX(1.D0,0.D0)
            DO 15 J = 1,10
15    B(I,J) = 0.D0
            DO 20 I = 1,18
            DO 20 J = 1,55
20    C(I,J) = 0.D0
            DO 25 I = 1,55
            DO 25 J = 1,3
            U(J) = 0.D0
25    G1(I,J) = 0.D0
            DO 30 I = 1,10
            DO 30 J = 1,3
30    G2(I,J) = 0.D0
            DO 40 I = 1,10
            DO 40 J = 1,18
            FTILDA(I,J) = 0.D0
40    F(I,J) = 0.D0
            DO 45 I = 1,200
45    WK(I) = 0.D0
            DO 50 I = 1,55
50    E(I) = DCMPLX(0.D0,0.D0)
            DO 55 I = 1,100
            DO 55 J = 1,100
55    X(I,J) = DCMPLX(0.D0,0.D0)
C
C-------------INITIALIZE INTEGERS---------------------------------------
C
            IRANKB = 0
            IRANKC = 0
C
C-----------READ IN SYSTEM DATA FROM  FILE 01 "RECONF" DATA-------------
C
            READ(1,100)NAME
            READ(1,110)((A(I,J),J=1,55),I=1,55)
            READ(1,100)NAME
            READ(1,110)((B(I,J),J=1,10),I=1,55)
            READ(1,100)NAME
            READ(1,110)((G1(I,J),J=1,3),I=1,55)
            READ(1,100)NAME
            READ(1,110)((C(I,J),J=1,55),I=1,18)
            READ(1,100)NAME
            READ(1,110)((F(I,J),J=1,18),I=1,10)
            READ(1,100)NAME
```

```
           READ(1.110)((G2(I.J).J=1.3).I=1.10)
           READ(1.100)NAME
           READ(1.110)((CLM(I.J).J=1.55).I=1.55)
100    FORMAT(1X.A4)
101    FORMAT(4I2)
110    FORMAT(4D20.13)
111    FORMAT(6E12.5)
C
C--------------DECISION QUE---------------------------------------------
C
           CALL FRTCMS('CLRSCRN ')
             WRITE(6.150)
150     FORMAT(/5X.'ENTER "1" TO WRITE UNDAMAGED EIGENSTRUCTURE.&  '
          *       ./5X.'           SYST. MATRICES.(READ "RECONF" DATA A1.'
          *       ./5X.'           "UNDIEG"."UNDABC".& "CLM".            '
          * ./5X.      'ENTER "2" TO READ UNDAMAGED  EIGENSTRUCTURE AND'
          *  ./5X.     '           WRITE SLOW RT SUBPACE DATA FOR USE BY'./5X.
          *            '           "ERSPACE" (WRITE "ERSPACE" DATA A1)  './5X.
          *            'ENTER "3" TO COMPUTE RECONFIGURED GAINS.        './5X.
          *            '           (WRITE "FRECON" DATA A1)             './5X.
          *            'ENTER "4" TO COMPUTE RECONFIG. EIGENSTRUCTURE  './5X.
          *            '           AND TIME RESPONSE (OPTMATD.OPTPLOT)  './5X.
          *            'ENTER "5" TO COMPUTE Q AND WRITE FILE 12        './5X.
          *            '           "EXACT" DATA.                        './)
           READ(*.155)K
155    FORMAT(I1)
160    FORMAT(I2)
           CALL FRTCMS('CLRSCRN ')
           IF(K.EQ.5)GOTO 272
           IF(K.EQ.4)GOTO 311
           IF(K.EQ.3)GOTO 170
           IF(K.EQ.2)GOTO 170
C------------STORE UND CL  IN UNITY MATRIX----------------------------
           DO 165 I = 1.55
           DO 165 J = 1.55
165    UNITY(I.J) = CLM(I.J)
           CALL EIGRF(CLM.55.55.1.E.X.100.WK.IER)
           WRITE(2.110)((X(I.J).J=1.55).I=1.55)
           WRITE(2.110)(E(I).I=1.55)
C      DO 156 I = 1.55
C      DO 156 J = 1.55
C      AA = A(I.J)
C156    ASING(I.J) = SNGL(AA)
C      DO 157 I = 1.55
C      DO 157 J = 1.10
C      BB = B(I.J)
C157    BSING(I.J) = SNGL(BB)
C      DO 158 I = 1.18
C      DO 158 J = 1.55
C      CC = C(I.J)
C158    CSING(I.J) = SNGL(CC)
C      WRITE(8.111)((ASING(I.J).J=1.55).I=1.55)
C      WRITE(8.111)((BSING(I.J).J=1.10).I=1.55)
C      WRITE(8.111)((CSING(I.J).J=1.55).I=1.18)
           WRITE(11.110)((UNITY(I.J).J=1.55).I=1.55)
C      WRITE(11.110)((G1(I.J).J=1.3).I=1.55)
C      WRITE(11.110)((G2(I.J).J=1.3).I=1.10)
           GOTO 9000
170    READ(2.110)((X(I.J).J=1.55).I=1.55)
           READ(2.110)(E(I).I=1.55)
           IF(K.EQ.3)GOTO 180
C
C             DECOUPLE LATERAL DYNAMICS FROM STABS
C             DECOUPLE LAT VECTORS FROM LONG ELEMENTS
177    WRITE(6.172)
172    FORMAT(/5X.'**** ENTER "1" TO MODIFY DESIRED EIGENSTRUCTURE ****'.
          *       /5X.'****              ELSE ENTER "0".****'.
          */)
           READ(*.155)IMOD
          CALL FRTCMS('CLRSCRN ')
          IF(IMOD.EQ.0)GCTO 180
          CALL FRTCMS('CLRSCRN ')
C     DO 178 I = 1.16
C     IF(I.LE.4)GOTO 178
C     IF(1.GE.9)GOTO 175
C     GOTO I78
C75   X(I.42) = DCMPLX(0.D0.0.D0)
C     X(I.52) = DCMPLX(0.D0.0.D0)
C     X(I.45) = DCMPLX(0.D0.0.D0)
C     X(I.41)= DCMPLX(0.D0.0.D0)
C78   CONTINUE
          DO 179 I = 5,8
C     X(I.38) = DCMPLX(0.D0.0.D0)
C     X(I.39) = DCMPLX(0.D0.0.D0)
C     X(I.43) = DCMPLX(0.D0.0.D0)
C     X(I.46) = DCMPLX(0.D0.0.D0)
          X(I.1) = DCMPLX(0.D0.0.D0)
          X(I.2) =  DCMPLX(0.D0.0.D0)
          X(I.3) = DCMPLX(0.D0.0.D0)
          X(I.4) = DCMPLX(0.D0.0.D0)
C     X(I.7) = DCMPLX(0.D0.0.D0)
C     X(I.8) =  DCMPLX(0.D0.0.D0)
C     X(I.17) = DCMPLX(0.D0.0.D0)
C     X(I.I8) = DCMPLX(0.D0.0.D0)
179   CONTINUE
C
C------------DISPLAY DESIRED EIGENSTRUCTURE IF REDUIRED--------------
C
180   WRITE(6.171)
I71   FDRMAT(/5X.'**** ENTER "I" TD DISPLAY DESIRED EIGENSTRCTURE****'.
        1 /5X,       '**** ENTER "0" OTHERWISE.                       ****'./
        2)
          READ(*.I55)IWRITE
          CALL FRTCMS('CLRSCRN ')
          IF(IWRITE.EQ.0)GOTD 255
          WRITE(6.200)
200   FORMAT(/5X.'**** UNDAMAGED (DESIRED) EIGENSTRUCTURE DF FI8 ****'./
        1)
          DO 250 I = 1.55.4
          WRITE(6.2I0)I..I+I.I+2.I+3.E(I).E(I+1).E(I+2).E(I+3)
210   FORMAT(/2X.'EIGENVALUE NO. '.I2.8X.4X.'EIGENVALUE NO. '.12.I2X.
        1'EIGENVALUE NO. '.12.12X.'EIGENVALUE NO. '.I2./2X.E12.5.IX.E12.5.
        2 2X.E12.5.1X.EI2.5.2X.E12.5.1X.EI2.5.2X.E12.5.1X.E12.5./)
21I   FORMAT(2X.E12.5.1X.E12.5.2X.E12.5.IX.E12.5.2X.E12.5.1X.E12.5.2X.
        1 E12.5.IX.E12.5./)
          WRITE(6.220)(X(J.I).X(J.I+1).X(J.I+2).X(J.I+3).J=1.55)
220   FORMAT(2X.E12.5.IX.E12.5.2X.E12.5.1X.E12.5.2X.E12.5.1X.E12.5,2X.
        IE12.5.IX.E12.5)
250   CONTINUE
C
C-------CDMPUTE X*DEL*X**-1 (UNDAMAGED/DESIRED EIGENSTRUCTURE)----------
C
255   DD 260 I = 1.55
          DEL(I.1) = E(I)
          DO 260 J = I.55
260   XS(I.J) = X(I.J)
          CALL CMATML(XS.DEL.55.55.55.XD)
          CALL LEQT1C(XS.55.55.UNITYC.55.55.0.WA1.IER)
          CALL CMATML(XD.UNITYC.55.55.55.XDX)
C
C--------------COMPUTE SVD DF 8 AND C (DAMAGED)------------------------
C
272   DD 270 I = I.55
          U8T(I.I) = 1.D0
          DD 270 J = 1.10
270   8S(I.J) = B(I.J)
          WRITE(6.271)
27I   FDRMAT(/5X.'**** CALLING LSVDF FOR 8 ****'./)
```

```
         CALL LSVDF(BS,55,55,10,UBT,55,55,SB,WKB,IER)
         WRITE(6,275)
275   FORMAT(/5X,'**** LSVDF OF "B" COMPLETE ****',/)
         CALL RANKO(BS,UBT,SB,55,10,UB0TC,UB1TC,BZINV,IRANKB,BGI)
         WRITE(6,276)IRANKB
276   FORMAT(5X,' **** RANK(B) = ',I2,'          ****')
C
         DO 280 I = 1,18
         UCT(I,I) = 1.00
         DO 280 J = 1,55
         CT(J,I) = C(I,J)
280   CS(I,J) = C(I,J)
         WRITE(6,281)
281   FORMAT(/5X,'**** CALLING LSVDF FOR C ****',/)
         CALL LSVDF(CS,55,18,55,UCT,55,18,SC,WKC,IER)
         WRITE(6,282)
282   FORMAT(/5X,'**** LSVDF OF "C" COMPLETE ****',/)
         CALL RANKO(CS,UCT,SC,18,55,UC0TC,UC1TC,CZINV,IRANKC,CGI)
         WRITE(6,283)IRANKC
283   FORMAT(5X, '**** RANK(C) = ',I2,'          ****')
         IF(K.EQ.3)GOTO 295
         IF(K.EQ.5)GOTO 312
C
C-------------WRITE SLOW EIGENSTRUCTURE---------------------------------
C
C        BUILD SLOW EIGENSPACE
286   ISLOW = 0
         KFAST = 0
         DO 285 I = 1,55
         ECHECK = DREAL(E(I))
         ECHECA = DABS(ECHECK)
C          FOR SYMMETRIC CASES USE Z < 8.5D-01=> 20 STATES
C--------------SYMMETRIC CASES------------------------------------------
C          FOR SEGMENT 1 :(24) USE Z < 7.59D-01=> 24 STATES
C          FOR SEGMENT 2 : 3.54D-01 < Z < 7.59D-01
C          FOR SEGMENT 3 : 0. < Z < 3.54D-01
         IF(ECHECA.LT.8.5D-01)GOTO 284
C     IF(ECHECA.LT.3.54D-01)GOTO 284
C     IF(ECHECA.LT.7.5900D-01)GOTO 284
C     IF(ECHECA.GT.3.54D-01)GOTO 284
         ISLOW = ISLOW + 1
C-------ISTATE = VECTOR OF INDICES CORRESPONDING TO SLOW STATES---------
         ISTATE(ISLOW) = I
         ESLOW(ISLOW) = E(I)
         ESING(ISLOW) = E(I)
         CALL SROOT(E(I),1,SR,.0125)
         WRITE(6,291)ISLOW,I,SR
291   FORMAT(1X,'SLOW STATE NO. ',I2,' UND STATE NO. ',I2,' S = ',E12.5,
     *1X,E12.5,'J')
         DO 285 J = 1,55
         XSLOW(J,ISLOW) = X(J,I)
         GOTO 285
284   KFAST = KFAST + 1
C-------KSTATE = VECTOR OF INDICES CORESPONDING TO FAST STATES---------
         KSTATE(KFAST) = I
285   CONTINUE
         IF(K.NE.1)GOTO 287
         WRITE(8,101)ISLOW
         WRITE(8,111)(ESING(I),I=1,ISLOW)
         GOTO 9000
287   IROWS = ISLOW
         WRITE(9,256)IROWS,KFAST,IRANKB,IRANKC
256   FORMAT(4I2)
         WRITE(9,110)(E(I),I=1,55)
         WRITE(9,110)((X(I,J),J=1,55),I=1,55)
         WRITE(9,257)(ISTATE(I),I=1,IROWS)
         WRITE(9,257)(KSTATE(I),I=1,KFAST)
257   FORMAT(40I2)
         WRITE(9,110)(ESLOW(I),I=1,IROWS)
```

```
          WRITE(9,110)((XSLDW(I,J),J=1,IRDWS),I=1,55)
          WRITE(9,110)((A(I,J),J=1,55),I=1,55)
          IROW = 55 - IRANKB
          WRITE(9,256)IROW
          WRITE(9,110)((UB1TC(I,J),J=1,55),I=1,IRDW)
          WRITE(9,110)((CT(I,J),J=1,18),I=1,55)
          GOTO 9000
C
C------------------FIND FTILDA (RECONFIGURED GAINS)--------------------
C
295   DO 300 I = 1,55
          DO 300 J = 1,55
          BR = BGI(I,J)
          BGIC(I,J) = DCMPLX(BR,0.D0)
          CR = CGI(I,J)
          CGIC(I,J) = DCMPLX(CR,0.D0)
          AR = A(I,J)
300   XDXA(I,J) = XDX(I,J) - DCMPLX(AR,0.D0)
          IF(K.EQ.5)GDTO 312
303   CALL CMATML(BGIC,XDXA,10,55,55,BUXD)
          CALL CMATML(BUXD,CGIC,10,55,18,FC)
          DO 310 I = 1,10
          DO 310 J = 1,18
C-------------FLAG IF COMPLEX GAINS ARE BEING COMPUTED------------------
          CHECK = DIMAG(FC(I,J))
          CHECKA = DABS(CHECK)
          IF(CHECKA.LT.1.D-03)GDTD 310
302   WRITE(6,301)I,J,CHECKA
301   FORMAT(/5X,'**** COMPLEX FEEDBACK GAIN (',I2,',',I2,') = ',E12.5)
310   FTILDA(I,J) = DREAL(FC(I,J))
          GDTD 314
C
C-------------WRITE FXACT FILE 12----------------------
C
312   WRITE(12,110)((UB1TC(I,J),J=1,55),I=1,IROW)
          GOTO 9000
C
C--------------WRITE FTILDA ON FILE 07 AND STOP------------------------
C
314   WRITE(7,110)((FTILDA(I,J),J=1,18),I=1,I0)
          GOTD 9000
C
C______________READ FTILDA FROM FILE 07 AND CONTINUE--------------------
C
311   READ(7,110)((FTILDA(I,J),J=1,18),I=1,10)
C
C--------------FIND EIGENSTRUCTURES OF RECONFIGURED SYSTEM--------------
C
          CALL VMULFF(FTILDA,C,10,18,55,10,18,FTC,10,IER)
          CALL VMULFF(B,FTC,55,10,55,55,10,BF,55,IER)
          DO 320 I = 1,55
          DO 320 J = 1,55
          CLMTIL(I,J) = A(I,J) + BF(I,J)
320   AF18(I,J) = CLMTIL(I,J)
          CALL EIGRF(CLMTIL,55,55,1,ETIL,XTIL,100,WKT,IER)
          CALL SRDOT(ETIL,55,ES,.0125)
C
C-------------FIND CLM FOR NOMINAL SYSTEM-------------------------------
C NOTE: USING CLMTIL FOR AF18 NOMINAL DUE TO STORAGE LIMITS
C       CLMTIL IS USED FOR MODAL FOLLOWING
          CALL VMULFF(F,C,10,18,55,10,18,FTC,10,IER)
          CALL VMULFF(B,FTC,55,10,55,55,10,BF,55,IER)
          DO 322 I = 1,55
          DO 322 J = 1,55
322   CLMTIL(I,J) = A(I,J) + BF(I,J)
C
C------------WRITE FILE 10 RECEIG DATA----------------------------------
C
          WRITE(10,110)((XTIL(I,J),J=1,55),I=1,55)
          WRITE(10,110)(ETIL(I),I=1,55)
C
C---------------DISPLAY RECONFIGURED EIGENSTRUCTURE------------
C
          WRITE(6,325)
325   FORMAT(/5X,'**** ENTER "1" TO DISPLAY RECONFIGURED         ****'
     1 ,/5X,     '**** EIGENSTRUCTURE. ELSE ENTER "0".            ****'
     2 ,/)
          READ(*,155)IDIS
          IF(IDIS.EQ.0)GOTO 400
326   WRITE(6,330)
330   FORMAT(/5X,'**** RECONFIGURED EIGENSTRUCTURE ****',/)
          DO 335 I = 1,55,4
          WRITE(6,210)I,I+1,I+2,I+3,ETIL(I),ETIL(I+1),ETIL(I+2),ETIL(I+3)
          WRITE(6,211)ES(I),ES(I+1),ES(I+2),ES(I+3)
          WRITE(6,220)(XTIL(J,I),XTIL(J,I+1),XTIL(J,I+2),XTIL(J,I+3),J=1,55)
335   CONTINUE
C      WRITE(6,336)
C336   FORMAT(/5X,'**** DAMAGED EIGENSTRUCTURE ****',/)
C      DO 337 I = 1,55,4
C      WRITE(6,210)I,I+1,I+2,I+3,EDAM(I),EDAM(I+1),EDAM(I+2),EDAM(I+3)
C      WRITE(6,220)(XDAM(J,I),XDAM(J,I+1),XDAM(J,I+2),XDAM(J,I+3),J=1,55
C337   CONTINUE
C
C------------------DISPLAY F AND FTILDA---------------------------------
C
400   WRITE(6,340)
340   FORMAT(/5X,'**** DAMAGED FEEDBACK GAIN MATRIX ****',//5X,' COL 1-9
     1',/)
          DO 345 I = 1,18,8
          WRITE (6,346)(F(J,I),F(J,I+1),F(J,I+2),F(J,I+3),F(J,I+4),F(J,I+5),
     1 F(J,I+6),F(J,I+7),F(J,I+8),J=1,10)
          IF(I.GT.2)GOTO 342
          WRITE(6,341)
341   FORMAT(/5X,'COL 10 - 18',/)
345   CONTINUE
346   FORMAT(2X,9(E12.5,1X))
342   WRITE(6,350)
350   FORMAT(/5X,'**** FTILDA ****',//5X,' COL 1-9',/)
          DO 355 I = 1,18,8
          WRITE (6,346)(FTILDA(J,I),FTILDA(J,I+1),FTILDA(J,I+2),FTILDA(J,I+3
     1),FTILDA(J,I+4),FTILDA(J,I+5),FTILDA(J,I+6),
     2 FTILDA(J,I+7),FTILDA(J,I+8),J=1,10)
          IF(I.GT.2)GOTO 401
          WRITE(6,351)
351   FORMAT(/5X,'COL 10 - 18',/)
355   CONTINUE
C
C----BUILD AF18,BF18,CF18,DF18 FOR RESPONSE CALCULATION AND PLOTTING----
C
401   WRITE(6,360)
360   FORMAT(/5X,'**** ENTER "1" FOR TIME RESPONSE CALC. ****',/5X,
     1            '**** ENTER "0" OTHERWISE.                ****',/)
          READ(*,155)IRESP
          IF(IRESP.EQ.0)GOTO 9000
402   NAF18(1) = 55
          NAF18(2) = 55
          CALL VMULFF(B,G2,55,10,3,55,10,BF18,55,IER)
          DO 405 I = 1,55
          DO 405 J = 1,3
405   BF18(I,J) = G1(I,J) + BF18(I,J)
          NBF18(1) = 55
          NBF18(2) = 3
          DO 410 I = 1,18
          DO 410 J = 1,55
410   CF18(I,J) = C(I,J)
          NCF18(1) = 18
          NCF18(2) = 55
          DO 415 I = 18
          DO 415 J = 1,3
415   DF18(I,J) = 0.D0
          NDF18(1) = 18
          NDF18(2) = 3
C----------------------------------------------------------------------
C
C  COMPUTE RESPONSE FOR 500 DATA POINTS AND CREATE OPTPLOT DATA FILE
C
C----------------------------------------------------------------------
          WRITE(6,500)
500   FORMAT(//1X,'COMPUTING TIME RESPONSE')
C-- WRITE OPTPLOT PARAMETERS AND NULL FEEDBACK MATRIX INTO DATA FILE --
          IORDER = 55
          NDUM4(1)= IORDER
          NDUM4(2)=3
          CALL NULL(DUM4,NDUM4)
          IFULL = NDUM4(1)*NDUM4(2)
C     WRITE(4,510) IORDER,3,501,1,1
C     WRITE(4,515) (DUM4(I),I=1,IFULL)
    510 FORMAT(5I5)
    515 FORMAT(5E14.7)
C-------- INITIALIZE THE STATE AND INPUT VARIABLS --------------------
          NXX(1)=55
          NXX(2)=1
          NU(1)=3
          NU(2)=1
          CALL NULL(XX,NXX)
          CALL NULL(U,NU)
C
C-----------------------COMPUTE RESPONSE-------------------------------
C
          WRITE(6,521)
521   FORMAT(/5X,'**** ENTER "1" FOR MODAL FOLLOWING ****',/5X,
     I           '**** ENTER "0" OTHERWISE.          ****',/)
          READ(*,155)IDAM
          CALL FRTCMS('CLRSCRN ')
          AMP = 1.
          NCONT = 1
          NST = 0
          NSTP = 240
          TS = .0125
          DO 520 I=0,500
          TIME=I*TS
C-------------- CHECK CONTROL PARAMETERS AND SET CONTROL INPUT ---------
          IF (I.GE.NST.AND.I.LE.NSTP) THEN
               U(NCONT)=AMP
          ELSE
               U(NCONT)=0.0
          END IF
          WRITE(4,530) TIME,(U(J),J=1,3),(XX(J),J=1,IORDER)
C............RESET SCALE FACTORS FOR X3,X4,X6,X7,X8.....................
C............TO RADIANS FOR PROPER CALCULATIONS.........................
          DO 800 K = 1,10
          IF(K.LT.3.OR.K.GT.8)GOTO 800
          IF(K.EQ.5)GOTO 800
          XX(K)=XX(K)/5.7296D+01
800   CONTINUE
C----------------- MULTIPLY AF18 BY XX(N)-------------------------------
          IORDER = 55
C------------------CHECK IF MODAL FOLLOWING IS REQD---------------------
          IF(IDAM.EQ.0)GOTO 525
          IF(XX(8).LT.0.0)GOTO 525
          DO 531 M = 1,IORDER
          DUM4(M) = 0.00
               DO 535 N = 1,IORDER
535   DUM4(M) = DUM4(M) + CLMTIL(M,N)*XX(N)
531   CONTINUE
          GOTO 545
525     DO 540 M=1,IORDER
```

```
                DUM4(M)=0.0
                    DO 550 N=1,IORDER
    550     DUM4(M)=DUM4(M)+AF18(M,N)*XX(N)
    540   CONTINUE
C----------------- MULTIPLY BF18 BY U(N) ------------------------------
545     DO 560 M=1,IORDER
                DUM5(M)=0.0
                    DO 570 N=1,3
    570     DUM5(M)=DUM5(M)+BF18(M,N)*U(N)
    560   CONTINUE
C------------- ADD AF18 XX(N) AND BF18 U(N) ---------------------------
                DO 580 M = 1,IORDER
                XX(M)=DUM4(M)+DUM5(M)
C.........SCALE XX3,XX4,XX6,XX7,XX8 TO DEGREES FOR PLOTTING PURPOSES....
C
                IF (M.LT.3.OR.M.GT.8) GOTO 580
                IF (M.EQ.5) GOTO 580
  222     XX(M) = XX(M)*5.7296D+01
    580 CONTINUE
    520 CONTINUE
    530 FORMAT(5E14.7)
C
C-----------------CREATE OPTMATD DATA FILE-----------------------------
C
C     CALL WRMATD(AF18,BF18,CF18,DF18,TS,IORDER)
            WRITE(6,999)
    999 FORMAT(//1X,'RESPONSE COMPLETE-OPMATD AND OPTPLOT DATA FILE CREATE
          *D')
            WRITE(6,1000)
  1000 FORMAT(//1X,'TO PLOT RESPONSE GO TO CONTROLS EXEC, SELECT ORACLS,
          *THEN SELECT OPTPLOT.',
          */1X,'YOU MUST BE AT A 618 TERMINAL')
9000  STOP
            END
C*********************************************************************
C
C              SUBROUTINE CMATML (COMPLEX MATRIX MULTIPLICATION)
C
C        COMPUTES: YY = AA * BB
C        IA = # OF ROWS IN AA
C        LL = # OF ROWS IN BB AND # OF COLUMNS IN AA
C        IB = # OF COLUMNS IN BB
C*********************************************************************
            SUBROUTINE CMATML(AA,BB,IA,LL,IB,YY)
            IMPLICIT REAL*8(A-H,O-Z)
            COMPLEX*16 AA(55,55),BB(55,55),YY(55,55)
            INTEGER IA,LL,IB
            DO 30 I = 1,IA
            DO 20 J = 1,IB
            YY(I,J) = (0.D0,0.D0)
            DO 10 INDEX = 1,LL
            YY(I,J) = YY(I,J) + AA(I,INDEX) * BB(INDEX,J)
10    CONTINUE
20    CONTINUE
30    CONTINUE
            RETURN
            END
C********************************************************************
C
C        SUBROUTINE RANKD: COMPUTES REQUIRED BLOCK STRUCTURES OF SVD
C                          OF "BS" FOR USE IN EIGENSTRUCTURE ANALYSIS
C                   CDR V.F. GAVITO
C                     JUNE 1986
C        NOTE: RZINVC IS NOT BEING COMPUTED !!!
C
C*********************************************************************
            SUBROUTINE RANKD(BS,UT,S,N,M,U0TC,U1TC,RZINVC,IRANK,BCI)
            IMPLICIT REAL*8(A-H,O-Z)
            REAL*8 BS(55,55),S(55),SM(55,55),RZ(55,55),RZINV(55,55),
     I U(55,55),UT(55,55),VT(55,55),SI(55),WK(50),SMI(55,55),WAREA(200)
     2,BGI(55,55),BSM(55,55)
      COMPLEX*16 RZINVC(55,55),U0TC(55,55),U1TC(55,55)
      INTEGER N,M,IRANK
      IRANK = M
      DO I I = I,55
      DO I J = 1,55
      BGI(I,J) = 0.D0
      RZ(I,J) = 0.D0
      RZINV(I,J) = 0.D0
      U0TC(I,J) = (0.D0,0.D0)
      U1TC(I,J) = (0.D0,0.D0)
      RZINVC(I,J) = (0.D0,0.D0)
      U(I,J) = 0.D0
      SM(I,J) = 0.D0
      SMI(I,J) = 0.D0
I     VT(I,J) = 0.D0
      DO 2 I = 1,50
2     WK(I) = 0.D0
      DO 3 I = I,M
      IF(S(I).GT.0.D0)GOTO 3
      IRANK = I-I
      GOTO 4
3     CONTINUE
4     DO 30 I = I,N
      DO 30 J = I,N
30    U(I,J) = UT(J,I)
      DO 40 I = I,N
      DO 40 J = I,IRANK
40    U0TC(J,I) = DCMPLX(U(I,J),0.D0)
      DO 50 I = 1,N
      DO 50 J = IRANK + I,N
50    UITC(J-IRANK,I) = DCMPLX(U(I,J),0.D0)
C     DO 60 I = I,M
      DO 60 I = 1,IRANK
      DO 60 J = I,M
60    VT(I,J) = BS(J,I)
      DO 70 I = 1,IRANK
70    SM(I,I) = S(I)
      CALL VMULFF(SM,VT,IRANK,M,M,55,55,RZ,55,IER)
C$$$$$$$$$$$$$$$$$$$$NOTE: RZINV IS NOT COMPUTED$$$$$$$$$$$$$$$$$$$$$$$$
C
C     CALL LGINF(RZ,10,IRANK,M,TOL,RZINV,10,SI,WK,IER)
C
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
      DO 3001 I = 1,IRANK
3001  SMI(I,I) = 1.D0/SM(I,I)
      CALL VMULFF(BS,SMI,M,IRANK,IRANK,55,55,RZINV,55,IER)
      DO 80 I = I,M
      DO 80 J = I,IRANK
80    RZINVC(I,J) = DCMPLX(RZINV(I,J),0.D0)
      CALL VMULFF(BS,SMI,M,IRANK,IRANK,55,55,BSM,55,IER)
      CALL VMULFF(BSM,UT,M,IRANK,N,55,55,BGI,55,IER)
      RETURN
      END
C********************** SUBROUTINE WRMATD ***************************
C
C         WRMATD WRITES THE 'AF18', 'BF18', 'CF18', AND 'DF18'
C     MATRICIES TO THE OPMATD DATA FILE. EXTRA NULL MATRICIES ARE ALSO
C     CREATED AND STORED TO CONFORM TO THE OPMATD DATA FILE STRUCTURE.
C     THESES MATRICIES ARE THE 'GAMD', 'FD', 'FK', 'CD', 'RD', 'V1',
C     'V2', AND 'SD'.
C
C*************************************************************************
      SUBROUTINE WRMATD(AD,BD,HD,GD,DELT,IORDER)
      IMPLICIT REAL*8(A-H,O-Z)
C
C     F18 SYSTEM MATRICIES
C
```

```
          DIMENSIDN AD(55,55),BD(55,3),HD(I4,55),GD(14,3)
C
C     EXTRA MATRICIES REQUIRED IN DPMATD DATA FILE
C
          DIMENSION FD(55,3),FK(55,14),OD(14,14),RD(3,3),V2(14,14),SD(55,14)
C
          DIMENSIDN NFD(2),NFK(2),NOD(2),NRD(2),NV2(2),NSD(2)
          INTEGER IDRDER
C------------ CREATE NULL MATRICIES TO BE STDRED DN DPMATO -----------
          NFD(1)=3
          NFD(2)=ICRDER
          NFK(1)=IORDER
          NFK(2)=14
          NQO(1)=14
          NOD(2)=I4
          NRO(1)=3
          NRD(2)=3
          NV2(1)=14
          NV2(2)=14
          NSD(I)=ICRDER
          NSD(2)=I4
C
          CALL NULL(FD,NFD)
          CALL NULL(FK,NFK)
          CALL NULL(QD,NOD)
          CALL NULL(RD,NRD)
          CALL NULL(V2,NV2)
          CALL NULL(SD,NSD)
C-------------- WRITE MATRICIES TD DPMATO DATA FILE-------------------
          I = 0
          IANS = 1
          IDOPTD = 1
          WRITE(3,150) I,IANS,IDDPTD
          WRITE(3,120) 1ORDER,3,14,0,1,DELT
          WRITE(3,131)
          WRITE(3,130) ((AD(I,J),J=I,IDRDER),I=1,IDRDER)
          WRITE(3,132)
          WRITE(3,130) ((BD(I,J),J=1,3),I=1,IORDER)
          WRITE(3,133)
          WRITE(3,130) ((HD(I,J),J=1,ICRDER),I=1,14)
          WRITE(3,135)
          WRITE(3,130) ((FD(I,J),J=1,IDRDER),I=1,3)
          WRITE(3,136)
          WRITE(3,130) ((FK(I,J),J=1,14),I=1,IDRDER)
          WRITE(3,137)
          WRITE(3,130) ((GD(I,J),J=1,3),I=1,14)
          WRITE(3,138)
          WRITE(3,130) ((QD(I,J),J=1,14),I=1,14)
          WRITE(3,I39)
          WRITE(3,130) ((RD(I,J),J=1,3),I=1,3)
          WRITE(3,14I)
          WRITE(3,130) ((V2(I,J),J=1,14),I=1,14)
          WRITE(3,I42)
          WRITE(3,130) ((SD(I,J),J=I,3),I=1,ICRDER)
C---------------------------------------------------------------------
120   FORMAT(5I5,5X,F10.5)
130   FORMAT(4020.13)
131   FORMAT(1X,2HAD)
132   FORMAT(1X,2HBO)
133   FORMAT(1X,2HHO)
134   FORMAT(1X,4HGAMO)
135   FORMAT(1X,2HFD)
136   FORMAT(1X,2HFK)
I37   FORMAT(IX,2HGD)
I38   FORMAT(1X,2HOD)
139   FORMAT(1X,2HRD)
140   FORMAT(1X,2HV1)
141   FORMAT(1X,2HV2)
142   FORMAT(1X,2HSD)
```

```
150   FORMAT(I1,3X,I1,3X,I1)
            RETURN
            END
C***********************************************************************
C
C             SUBROUTINE SROOT: CONVERTS Z-PLANE ROOTS TO
C                               S-PLANE ROOTS
C
C                   CDR V.F. GAVITO   JULY 1986
C
C***********************************************************************
            SUBROUTINE SROOT(Z,N,S,T)
            IMPLICIT REAL*8(A-H,O-Z)
            COMPLEX*16 Z(55),S(55)
            INTEGER N
            DO 10 I = 1,N
            RS = DIMAG(Z(I))
            RC = DREAL(Z(I))
            B = (DATAN2(RS,RC))/T
            R2 = CDABS(Z(I))
            A = -DLOG(R2)/T
            S(I) = DCMPLX(-A,B)
10    CONTINUE
            RETURN
            END
```

APPENDIX E
ERSPACE

```
C************************************************************************
C
C       ERSPACE: TESTS IF DESIRED/UNDAMAGED SLOW RIGHT EIGENVECTORS
C               ARE MEMBERS OF THE DAMAGED RIGHT HAND NULL SPACES.
C               AND WRITES DATA FILE FOR USE BY ELSPACE
C
C                    CDR V.F. GAVITO
C                       JULY 1986
C
C               FILE 01 = ERSPACE DATA
C                         (READ)
C               FILE 08 = ELSPACE DATA (LEFT EIGENSPACE DATA)
C                         (WRITE)
C************************************************************************
            IMPLICIT REAL*8(A-H.O-Z)
            DIMENSION A(55.55).RESR(24).CT(55.55).WA(55).UNITY(55.55)
            COMPLEX*16 X(55.55).E(55).XS(55.55).T(55.55).TSLOW(55.24)
            COMPLEX*16 U1TC(55.55).AML(55.55).EIGD(24).VD(55.24).Z1,V(55.24)
            COMPLEX*16 UNITYC(55,55).RNULLC(55,55).WV(55),WV1C(55),
          1           RES(45,24),ESLOW(24).XSLOW(55,24),Z2
            REAL*4 SWV(110).VL8(1I0).VU8(110).G(1).RA(111,111),
          1       WKS(9000).ASING(55,55).CTSING(55,18).ERS,ECS.XRS,XCS,
          2        TRS.TCS.URS.UCS
            COMPLEX*8 ESING(24).TSLING(55,24).XSLING(55,24).XSING(55.55)
            COMPLEX*8 EING(55),U1BT(55.55)
            INTEGER N1.M1.IELE,IWK(1000).IC(110),IDG(1),IY.IRANK,IROW(55.24)
            INTEGER DF(111),ISTATE(55),KSTATE(55),IROWS,KFAST,IAR8(55).IUCT1
C
C--------------READ FILE 01 (ERSPACE DATA)------------------------------
C
            READ(1,2)IROWS,KFAST,IRANKB,IRANKC
            READ(1,3)(E(I),I=1.55)
            READ(1,3)((X(I,J).J=1.55),I=I,55)
            READ(1.4)(ISTATE(I),I=1,IROWS)
            READ(1.4)(KSTATE(I).I=1.KFAST)
            READ(1,3)(ESLOW(I).I=1.IROWS)
            READ(1.3)((XSLOW(I.J).J=1.IROWS),I=1.55)
            READ(1,3)((A(I.J),J=1,55),I=1,55)
            READ(1.2)IU81
            READ(1,3)((U1TC(I,J).J=1,53),I=I,IU8I)
            READ(1.3)((CT(I,J).J=1,18).I=1,55)
2     FORMAT(4I2)
3     FORMAT(4D20.13)
4     FORMAT(40I2)
5     FORMAT(6E12.5)
C
C------------------INITIALIZE CONSTANTS-------------------------------
C
            CALL FRTCMS('CLRSCRN ')
            N1 = 55
            TSTEP = 1.25D-02
            DO 6 I = 1.N1
            DO 6 J = 1.N1
            ASING(I,J) = SNGL(A(I,J))
            UNITY(I,J) = 0.D0
6     UNITYC(I,J) = DCMPLX(0.D0,0.D0)
            DO 10 I = 1,N1
            UNITY(I,I) = 1.D0
            UNITYC(I.I) = DCMPLX(I.D0.0.D0)
            DO 10 J = 1,IROWS
10    V(I,J) = XSLOW(I,J)
```

```
          ICONJG = 0
C
C--------DISPLAY SLOW EIGENVALUE INDICES FOR ARBITRARY ELEMENT----------
C-----------------------SELECTION-------------------------------------
C
          WRITE(6,11)
11    FORMAT(/5X,'**** INDICES OF SLOW EIGENVALUES ****',/)
          WRITE(6,12)(ISTATE(I),I=1,IROWS)
12    FORMAT(5(2X,I2,2X),/)
C     WRITE(6,14)
C4    FORMAT(/5X,'**** ENTER NO. OF ARBITRARY EIGENVECTOR ****',/5X,
C    1          '****           FOR ADS.                ****',/)
C     READ(*,2)IELE
C     WRITE(6,I3)IELE
C3    FORMAT(/5X,'**** ENTER ROW NOS. OF THE ',I2,' ARBITRARY    ****',
C    I/5X,      '**** ELEMENTS IN I2 FORMAT.                   ****',/)
C     READ(*,4)(IARB(I),I=1,IELE)
C-----------FOR RECR25 THERE ARE 27 ARB ELEMENTS NOT INCLUD STABS------
C     IELE = 36
          IELE = 27
C     DO I3 M = 1,28
C3    IARB(M) = M + 8
C     DO 14 M = 29,34
C4    IARB(M) = M + 9
C     IARB(35) = 45
C     IARB(36) = 47
          DO I3 M = I,27
I3    IARB(M) = M+I6
I70   DO 55 I = I,IROWS
          ZI = ESLOW(I)
          DO 56 J = 1,NI
          DO 56 K = I,NI
56    AML(J,K) = DCMPLX(A(J,K),0.D0) - ZI*UNITYC(J,K)
          IY = IUBI
44    CALL CMATML(UITC,AML,IY,NI,NI,RNULLC)
          DO 57 J = 1,NI
57    WV(J) = V(J,I)
59    CALL CMATML(RNULLC,WV,IY,NI,1,WV1C)
          DO 58 J = 1,IY
58    RES(J,I) = WV1C(J)
C        FIND 2-NORM OF THE RESIDUAL
          Y = 0.D0
          DO 76 J = I,IY
76    Y = Y + DREAL(RES(J,I))**2+ DIMAG(RES(J,I))**2
          ROBJ = DSQRT(Y)
          RESR(I) = ROBJ
C        IF 2-NORM OF RESIDUAL IS OK -- JUMP OUT
          IF(ROBJ.LE.I.D00)GOTO 55
C     GOTO 55
C**********************************************************************
C
C        IF EIGENVALUE IS COMPLEX, THEN PERFORM THE OPTIMIZATION
C        ONE TIME FOR BOTH THE EIGENVALUE AND ITS CONJUGATE.
C
C**********************************************************************
96I   IF(DIMAG(ZI).EQ.0.D0)GOTO 960
          IF(ICONJG.EQ.1)GOTO 97
          ICONJG = I
C---------INITIALIZE ARBITRARY ELEMENTS (FAST ELEMENTS)----------------
960   DO 100 II = 1,IELE
100   IROW(II,I) = IARB(II)
C**********************************************************************
C
C        SET UP FOR ADS OPTIMIZER CALL IN SINGLE PRECISION FOR RESIDUAL
C
C**********************************************************************
962   SROBJ = SNGL(ROBJ)
          INFO = 0
          ISTRAT = 0
```

```
      IOPT = 3
      IONED = 1
      NOV = 2*N1
      NCON = 0
      IGRAD = 0
      IPRINT = 1000
      NGT = 0
      NRA = 55
      NCOLA = 110
      CALL SROOT(Z1,Z2,TSTEP)
      WRITE(6,963)I,Z1,Z2,ISTATE(I)
963   FORMAT(/5X,'**** ENTERING ADS FOR ESLCW(',I2,') Z = ',E12.5,1X,
     1 E12.5,'J  ****',/5X,'****',28X,'S = ',E12.5,1X,E12.5,'J',/5X,
     2 '**** UNDAMAGED EIGENVECTOR NO. ',I2,/)
C        SET BOUNDS
C        ASK ALLOWED TOLERANCE ON DESIGN EIGENVECTORS
      IF(IELE.EQ.0)GOTO 1007
1008  WRITE(6,1003)
1003  FORMAT(/5X,'**** ENTER REAL UPPER BOUND OF ARBITRARY ELEMENTS ****
     1',/5X,  '**** IN "F12.5" FORMAT. ENTER "1" FOR NO BOUND.   ****')
      READ(*,1020)UR
      IF(DIMAG(Z1).NE.0.D0)GOTO 1015
      UC = 0.
      GOTO 1016
1015  WRITE(6,1004)
1004  FORMAT(/5X,'**** ENTER IMAG UPPER BOUND OF ARBITRARY ELEMENTS ****
     1',/5X,  '**** IN "F12.5" FORMAT. ENTER "1" FOR NO BOUND.   ****')
      READ(*,1020)UC
1016  IF(UR.NE.1.E+06)GOTO 1006
      UR = 0.1E+21
1006  IF(UC.NE.1.E+06)GOTO 1005
      UC = 0.1E+21
1005  IF(IELE.EQ.N1)GOTO 1013
1007  WRITE(6,1010)
1010  FORMAT(/5X,'**** ENTER ALLOWED EIGENVECTOR TOLERANCE IN "F12.5" FO
     1RMAT ****',/5X,'**** FOR THOSE EIGENVECTOR ELEMENTS      ****',/5X,
     2              '**** WHICH ARE NOT ARBITRARY.            ****',/)
      WRITE(6,1011)
1011  FORMAT(5X,'**** ENTER TOLERANCE ON REAL PART ****',/)
      READ(*,1020)EPSR
1020  FORMAT(F12.5)
      IF(DIMAG(Z1).NE.0.D0)GOTO 1017
      EPSC = 0.
      GOTO 1013
1017  WRITE(6,1012)
1012  FORMAT(5X,'**** ENTER TOLERANCE ON IMAG PART ****',/)
      READ(*,1020)EPSC
1013  CALL FRTCMS('CLRSCRN ')
      JJ = 0
      DO 77 J = 1,NOV-1,2
      JJ = JJ + 1
      KFIND = 0
      IF(IELE.EQ.0)GOTO 1026
1014  DO 1025 K = 1,IELE
      IF(JJ.NE.IROW(K,I))GOTO 1025
      KFIND = 1
      VUB(J) = UR
      VUB(J+1) = UC
      VLB(J) = -UR
      VLB(J+1) = -UC
1025  CONTINUE
C--------NOTE: PRESENTLY THE ENTIRE VECTOR IS ALLOWED TO VARY BY AN----
C             AMOUNT EQUAL TO EPSR,EPSC. INPUT EPSR=EPSC=0.0CURRENTLY
C     IF(KFIND.EQ.1)GOTO 1027
1026  XX = DREAL(XSLOW(JJ,I))
      YY = DIMAG(XSLOW(JJ,I))
      VUB(J) = SNGL(XX) + EPSR
      VUB(J+1) = SNGL(YY) + EPSC
      VLB(J) = SNGL(XX) - EPSR
```

```
          VLB(J+1) = SNGL(YY) - EPSC
1027  X1 = DREAL(WV(JJ))
          Y1 = DIMAG(WV(JJ))
          SWV(J) = SNGL(X1)
          SWV(J+1) = SNGL(Y1)
          IC(J) = 0
          IC(J+1) = 0
          DF(J) = 0
          DF(J+1) = 0
77    CONTINUE
84    CALL ADS(INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,SWV,VLB,
         1 VUB,SROBJ,G,IDG,NGT,IC,DF,RA,NRA,NCOLA,WK3,9000,IWK,1000)
          JJ = 0
          DO 78 J = 1,NDV-1,2
          JJ = JJ + 1
          WV(JJ) = CMPLX(SWV(J),SWV(J+1))
78    V(JJ,I) = WV(JJ)
          IF(INFO.EQ.0)GOTO 54
C        REEVALUATE OBJECTIVE FUNCTION
          CALL CMATML(RNULLC,WV,IY,N1,1,WV1C)
          DO 82 J = 1,IY
82    RES(J,I) = WV1C(J)
          Y = 0.D0
          DO 83 J = 1,IY
83    Y = Y + DREAL(RES(J,I))**2 + DIMAG(RES(J,I))**2
          ROBJ = DSQRT(Y)
          RESR(I) = ROBJ
          SROBJ = SNGL(ROBJ)
C        CONTINUE WITH ADS OPTIMIZATION (MINIMIZING ROBJ)
          GOTO 84
54    WRITE(6,95)ROBJ,I
95    FORMAT(/1X,'**** OBJECTIVE FUNCTION VALUE = ',E10.3,' ****',/1X,
         1           '**** FOR EIGENVECTOR NO. ',I2,'              ****',/)
          IF(DIMAG(Z1).EQ.0.D0)GOTO 55
          IF(ICONJG.EQ.1)GOTO 55
97    DO 98 K = 1,N1
          ICONJG = 0
98    XSLOW(K,I) = DCONJG(V(K,I-1))
55    CONTINUE
C***********************************************************************
C------DISPLAY RESULTS OF <UIT*<A-LAMDHA*I>>*DESIRED BASE EIGENVECTORS--
C***********************************************************************
          CALL FRTCMS('CLRSCRN ')
          WRITE(6,65)
65    FORMAT(//5X,'**** THE FOLLOWING VECTOR DISPLAYS THE NEARNESS OF  *
         1***',/5X,  '**** EACH SLOW EIGENVECTOR TO THE ALLOWABLE RIGHT    *
         2***',/5X,  '**** HAND SLOW EIGENSPACE FOR THE DAMAGED F18 MODEL. *
         3***',/)
          WRITE(6,60)(RESR(I),I=1,IROWS)
60    FORMAT(2X,D20.13)
          WRITE(6,150)
150   FORMAT(/5X,'**** ENTER "1" IF RESIDUAL IS UNSAT TO RENTER ADS ****
         1 ',/5X,    '**** ELSE ENTER "0".                              ****
         2',/)
          READ(*,155)IOPT
155   FORMAT(I1)
          CALL FRTCMS('CLRSCRN ')
          IF(IOPT.EQ.0)GOTO 160
          DO 165 J = 1,IROWS
          DO 165 I = 1,55
165   XSLOW(I,J) = V(I,J)
          GOTO 170
160   DO 93 K = 1,IROWS
          DO 93 J = 1,55
          KK = ISTATE(K)
93    X(J,KK) = V(J,K)
C
C--------------FIND LEFT HAND EIGENVECTORS-----------------------------
C
```

```
            DO 175 I = 1,55
            DO 175 J = 1,55
175   XS(I,J) = X(I,J)
            CALL LEOT1C(XS,55,55,UNITYC,55,55,0,WA,IER)
            DO 180 I = 1,55
            DO 180 J = 1,55
180   T(J,I) = UNITYC(I,J)
C         REINIT UNITYC
            DO 181 I = 1,N1
            DO 181 J = 1,N1
181   UNITYC(I,J) = DCMPLX(UNITY(I,J),0.D0)
C
C-------------BUILD SLOW LEFT HAND EIGENVECTORS-------------------------
C
            DO 185 J = 1,IROWS
            DO 185 I = 1,55
            KK = ISTATE(J)
185   TSLOW(I,J) = T(I,KK)
C
C------------WRITE FILE 08 FOR USE BY ELSPACE IN SINGLE PRECISION------
C
            WRITE(6,198)
198   FORMAT(/5X,'**** WRITING ELSPACE DATA FILE 08 ****',/)
            DO 190 I = 1,55
            ER = DREAL(E(I))
            EC = DIMAG(E(I))
            ERS = SNGL(ER)
            ECS = SNGL(EC)
            EING(I) = CMPLX(ERS,ECS)
            DO 190 J = 1,55
            XR = DREAL(X(I,J))
            XC = DIMAG(X(I,J))
            XRS = SNGL(XR)
            XCS = SNGL(XC)
            XSING(I,J) = CMPLX(XRS,XCS)
190   ASING(J,K) = SNGL(A(J,K))
            DO 195 J = 1,55
            DO 195 K = 1,18
195   CTSING(J,K) = SNGL(CT(J,K))
            DO 196 I = 1,IROWS
            ER = DREAL(ESLOW(I))
            EC = DIMAG(ESLOW(I))
            ERS = SNGL(ER)
            ECS = SNGL(EC)
            ESING(I) = CMPLX(ERS,ECS)
            DO 196 J = 1,55
            TR = DREAL(TSLOW(J,I))
            TC = DIMAG(TSLOW(J,I))
            TRS = SNGL(TR)
            TCS = SNGL(TC)
            XR = DREAL(XSLOW(J,I))
            XC = DIMAG(XSLOW(J,I))
            XRS = SNGL(XR)
            XCS = SNGL(XC)
            XSLING(J,I) = CMPLX(XRS,XCS)
196   TSLING(J,I) = CMPLX(TRS,TCS)
            DO 197 I = 1,IUB1
            DO 197 J = 1,55
            UR = DREAL(U1TC(I,J))
            UC = DIMAG(U1TC(I,J))
            URS = SNGL(UR)
            UCS = SNGL(UC)
197   U1BT(I,J) = CMPLX(URS,UCS)
            WRITE(8,2)IROWS,IUB1
            WRITE(8,4)(ISTATE(I),I=1,IROWS)
            WRITE(8,5)(ESING(I),I=1,IROWS)
            WRITE(8,5)((ASING(I,J),J=1,55),I=1,55)
            WRITE(8,5)((TSLING(I,J),J=1,IROWS),I=1,55)
            WRITE(8,5)((XSLING(I,J),J=1,IROWS),I=1,55)
```

```
          WRITE(8,5)((CTSING(I,J),J=1,18),I=1,55)
          WRITE(8,5)(EING(I),I=1,55)
          WRITE(8,5)((XSING(I,J),J=1,55),I=1,55)
          WRITE(8,5)((U1BT(I,J),J=1,55),I=1,IUB1)
66    STOP
          END
C**********************************************************************
C
C              SUBROUTINE CMATML (COMPLEX MATRIX MULTIPLICATION)
C
C       COMPUTES: YY = AA * BB
C       IA = # OF ROWS IN AA
C       LL = # OF ROWS IN BB AND # OF COLUMNS IN AA
C       IB = # OF COLUMNS IN BB
C**********************************************************************
          SUBROUTINE CMATML(AA,BB,IA,LL,IB,YY)
          COMPLEX*16 AA(55,55),BB(55,55),YY(55,55)
          INTEGER IA,LL,IB
          DO 30 I = 1,IA
          DO 20 J = 1,IB
          YY(I,J) = (0.D0,0.D0)
          DO 10 INDEX = 1,LL
          YY(I,J) = YY(I,J) + AA(I,INDEX) * BB(INDEX,J)
10    CONTINUE
20    CONTINUE
30    CONTINUE
          RETURN
          END
C**********************************************************************
C
C             SUBROUTINE SROOT: CONVERTS Z-PLANE ROOTS TO
C                               S-PLANE ROOTS
C
C                   CDR V.F. GAVITO   JULY 1986
C
C**********************************************************************
          SUBROUTINE SROOT(Z,S,T)
          IMPLICIT REAL*8(A-H,P-Z)
          COMPLEX*16 Z,S
          RS = DIMAG(Z)
          RC = DREAL(Z)
          B = (DATAN2(RS,RC))/T
          R2 = CDABS(Z)
          A = -DLOG(R2)/T
          S = DCMPLX(-A,B)
          RETURN
          END
```

APPENDIX F

ELSPACE

```
C**********************************************************************
C
C       ELSPACE: TESTS IF RT/LT SLOW LEFT EIGENVECTORS
C                RESULTING FROM ERSPACE OPTIMIZATION ARE MEMBERS OF THE
C                DAMAGED NULL SPACES.
C
C       NOTE: COMPUTATIONS ARE IN SINGLE PRECISION
C
C                    CDR V.F. GAVITO
C                    NOVEMBER 1986
C
C               FILE 01 = ELSPACE DATA
C                         (READ)
C               FILE IPASS = ADSDAT DATA   OR   XSLOW DATA
C                          (WRITE CR READ)(WRITE)
C               FILE 04 = LRES DATA
C                         (WRITE UNDAMAGED) (REAL OAMAGED)
C               FILE 07 = NULL DATA
C                         (READ)
C               FILE 0B = OPTEIG DATA
C                         (WRITE)
C               FILE 09 = SEGEIG1 DATA (SEGMENT 1 DESIGN DATA)
C               FILE I0 = SEGEIG2 DATA (SEGMENT 2 OESIGN DATA)
C               FILE I1 = SEGEIG3 DATA (SEGMENT 3 DESIGN DATA)
C                         (WRITE)
C                      NORMS
C               RESRS()/RESLS() = SUM OF VECTOR COMPCNENTS (COMPLEX)
C               RESR2()/RESL2() = 2-NCRMS (REAL)
C               RESR1()/RESL1() = 1-NORMS (REAL)
C               REINFR()/REINFL() = INFINITY NORM (REAL)
C               INFNRR()/INFNRL() = ROW NUMBER CF INFINITY NORM
C               ELUND()/ELDAM() = LEFT NULL RESIDUAL VECTORS
C               DTNORM() = 2-NCRM OF DIFFERENCE BETWEEN LEFT EVECTORS
C
C**********************************************************************
      IMPLICIT REAL*4(A-H.O-Z)
      REAL*4 LTEST
      DIMENSION A(55,55),CT(55,55),WA(55),UNITY(55,55).DTNORM(24)
      DIMENSION SCT(55).WKCT(110).UCTT(55.55),RESR2(24),RESL2(24)
      DIMENSION G(1).OF(111).RA(111,222).WK(10000),XX(111),VU8(I11)
      DIMENSION VLB(111),RESR1(24),RESL1(24),REINFR(24),REINFL(24)
      COMPLEX*8 TSLOW(55.24).AML(55.55).UNITYC(55.55).UCT1TC(55.55)
      COMPLEX*8 XSLOW(55.24).UB1TC(55.55).RESR(24),RESL(24)
      COMPLEX*8 UCA(55,55),LNULL(24.37.55).LOBJ(55).ESLOW(24)
      COMPLEX*8 U8A(55.55).RNULL(24,45.55),RO8J(55).E(55).X(55,55)
      COMPLEX*8 ATML(55.55).WVL(55).NML(55.55).T(55,55).XS(55.55)
      CCMPLEX*8 WVR(55).NMR(55.55),RR.RL.Z1.Z2.Z3.Z4.Z5.ROT.SCHG
      CCMPLEX*8 RUNULL(1.45.55).LUNULL(I.37.55).XT(55.55).XSAVE(55.55)
      COMPLEX*8 RESRR(24).RESLL(24).O8JC.Z5A.RESRS(24).RESLS(24)
      COMPLEX*B ELUND(37,24),ELDAM(37.24).O1FF(24)
      COMPLEX*B TDES(55,24),DT(55)
      COMPLEX*16 EO(55),XD(55.55).EDR(55),XDR(55,55)
      REAL*8 DR.DC
      INTEGER IROWS.IRANCT.IUBI.IDG(1).IC(1).IWK(1200).IO(8).ISTATE(55)
      INTEGER INULL.IARB(55),ICHG.INP,IADS.ISEG.IEEN.I1MAG.IRECHK
      INTEGER INFNRR(24).IFFNRL(24).IUND
C--------------INIT COMPLEX MATRICES-----------------------------------
      DO 600 I = 1,55
      DO 600 J = 1,55
      UCT1TC(1,J) = CMPLX(0.,0.)
      UB1TC(I.J) = CMPLX(0.,0.)
```

```
          UBA(I,J) = CMPLX(0.,0.)
          UCA(I,J) = CMPLX(0.,0.)
600   CONTINUE
C
C--------------READ FILE 01 (ELSPACE DATA)-----------------------------
C
          READ(1,2)IROWS,IUB1
          READ(1,4)(ISTATE(I),I=1,IROWS)
          READ(1,3)(ESLOW(I),I=1,IROWS)
          READ(1,3)((A(I,J),J=1,55),I=1,55)
          READ(1,3)((TSLOW(I,J),J=1,IROWS),I=1,55)
          READ(1,3)((XSLOW(I,J),J=1,IROWS),I=1,55)
          READ(1,3)((CT(I,J),J=1,18),I=1,55)
          READ(1,3)(E(I),I=1,55)
          READ(1,3)((X(I,J),J=1,55),I=1,55)
          READ(1,3)((UB1TC(I,J),J=1,55),I=1,IUB1)
2     FORMAT(4I2)
3     FORMAT(6E12.5)
4     FORMAT(40I2)
8     FORMAT(I1)
C
C--------------INIT CONSTANTS------------------------------------------
C
          N1 = 55
          TSTEP = 0.0125
          INFO = 0
          ITER = 0
          IADS = 0
          IDG(1) = 0
          G(1) = 0.
          IC(1) = 0
          INULL = 0
          ICHG = 0
          IIMAG = 0
          INULLC = 0
          INP = 0
          IRECHK = 0
          IUND = 0
          WEIGHT = 0.01
          RLU = 0.
          RLD = 0.
          RLREAL = 0.
          IRANCT = 18
          DO 5 I = 1,55
          IARB(I) = 0
          ROBJ(I) = CMPLX(0.,0.)
          LOBJ(I) = CMPLX(0.,0.)
          DO 5 J = 1,55
          XSAVE(I,J) = X(I,J)
          XS(I,J) = X(I,J)
          UNITY(I,J) = 0.
          UCTT(I,J) = 0.
          UNITYC(I,J) = CMPLX(0.0,0.0)
5     CONTINUE
          DO 6 I = 1,55
          ED(I) = E(I)
          UNITY(I,I) = 1.0
          UNITYC(I,I) = CMPLX(1.0,0.0)
6     CONTINUE
          DO 7 I = 1,24
          RESRS(I) = CMPLX(0.,0.)
          RESLS(I) = CMPLX(0.,0.)
          RESRR(I) = CMPLX(0.,0.)
          RESLL(I) = CMPLX(0.,0.)
          RESR2(I) = 0.
          RESL2(I) = 0.
          RESR1(I) = 0.
          RESL1(I) = 0.
          REINFR(I) = 0.
```

```
            REINFL(I) = 0.
            RESR(I) = CMPLX(0.,0.)
            RESL(I) = CMPLX(0.,0.)
            DO 7 J = 1,37
            ELUND(J,I) = CMPLX(0.,0.)
            ELDAM(J,I) = CMPLX(0.,0.)
7      CONTINUE
C
C-------------SAVE DESIRED SLOW LEFT EIGENVECTORS FOR OPTIMIZATION-----
C
            DO 660 I = 1,55
            DO 660 J = 1,IROWS
660    TDES(I,J) = TSLOW(I,J)
C
C-------------FIND DESIRED LEFT EIGENVECTOR MATRIX T--------------------
C
620    CALL LEOT1C(XS,55,55,UNITYC,55,55,0,WA,IER)
            DO 9 I = 1,55
            DO 9 J = 1,55
9      T(J,I) = UNITYC(I,J)
C------------- REINIT COMPLEX IDENTITY & XS-----------------------------
            DO 11 I = 1,55
            DO 11 J = 1,55
            XS(I,J) = CMPLX(0.,0.)
            U = UNITY(I,J)
11     UNITYC(I,J) = CMPLX(U,0.0)
            IF(IRECHK.EQ.0)GOTO 17
            DO 640 I = 1,55
            DO 640 J = 1,IROWS
            KK = ISTATE(J)
640    TSLOW(I,J)=T(I,KK)
            GOTO 208
C
C-------------FIND SVD OF C**T (DAMAGED)-------------------------------
C
17     DO 10 I = 1,55
10     UCTT(I,I) = 1.0
            CALL LSVDF(CT,55,55,18,UCTT,55,55,SCT,WKCT,IER)
            DO 15 I = 1,18
            IF(SCT(I).GT.0.0)GOTO 15
            IRANCT = I-1
            GOTO 20
15     CONTINUE
20     CALL FRTCMS('CLRSCRN ')
            WRITE(6,16)IRANCT
16     FORMAT(/5X,'**** RANK(C**T) = ',I2,' ****',/)
            DO 25 I = 1,55
            DO 25 J = IRANCT + 1,55
            UCX = UCTT(J,I)
25     UCTITC(J-IRANCT,I) = CMPLX(UCX,0.0)
C------------------ANALYSIS INFO---------------------------------------
            WRITE(6,18)
18     FORMAT(/5X,'**** ENTER "1" IF PASS IS FOR UNDAM ANALYSIS ****',
     *        /5X,'**** ENTER "0" ELSE.                           ****',/)
            READ(*,8)IUND
            IF(IUND.EQ.1)GOTO 19
            READ(4,3)((ELUND(I,J),J=1,37),I=1,IROWS)
C
C--------------PASS OUE------------------------------------------------
C
19     WRITE(6,41)
41     FORMAT(/5X,'**** ENTER "1" IF FIRST PASS THRU ELSPACE ****',/5X,
     1           '**** ENTER PASS NUMBER OTHERWISE.          ****',/)
            READ(*,8)IPASS
            IF(IPASS.EQ.1)GOTO 42
            READ(IPASS,2)IROWS,IPASSA
            READ(IPASS,4)(ISTATE(I),I=1,IROWS)
            READ(IPASS,3)((X(I,J),J=1,55),I=1,55)
            READ(IPASS,3)((T(I,J),J=1,55),I=1,55)
```

```
          READ(IPASS.3)((XSLOW(I.J).J=1.IROWS).I=I.55)
          READ(IPASS.3)((TSLOW(I.J).J=1.IROWS).I=I.55)
          READ(IPASS.3)(ESLOW(I).I=I.IROWS)
C
C-------------REINIT E AFTER ELSPACE EXECUTION-------------------------
C
42    DO 27 I = 1.IROWS
          II = ISTATE(I)
          ED(II) = ESLOW(I)
27    E(II) = ESLOW(I)
C
C-------------REINIT XD AFTER ELSPACE EXECUTION------------------------
C
          DO 28 I = I.55
          DO 28 J = I.55
28    XO(I.J) = X(I.J)
C
C-------------FIND NULL SPACE OPERATORS--------------------------------
C
26    DO 30  I = I.IROWS
          Z1 = ESLOW(I)
          IF(ICHG.EQ.0)GOTO 34
          Z1 = E(ICHG)
34    DO 35 J = I.55
          DO 35 K = I.55
          AR = A(J.K)
          ATR = A(K.J)
          AML(J.K) = CMPLX(AR.0.0) - Z1*UNITYC(J.K)
35    ATML(J.K) = CMPLX(ATR.0.0) - ZI*UNITYC(J.K)
          CALL CMATML(UBITC.AML.IUBI.55.55.UBA)
          CALL CMATML(UCTITC.ATML.55-IRANCT.55.55.UCA)
          DO 36 JJ = I.IUBI
          DO 36 KK = I.55
          IF(ICHG.EQ.0)GOTO 37
          RNULL(INULLC.JJ.KK) = UBA(JJ.KK)
          GOTO 36
37    RNULL(I.JJ.KK) = UBA(JJ.KK)
36    CONTINUE
          DO 40 JJ = I.55-IRANCT
          DO 40 KK = I.55
          IF(ICHG.EQ.0)GOTO 39
          LNULL(INULLC.JJ.KK) = UCA(JJ.KK)
          GOTO 40
39    LNULL(I.JJ.KK) = UCA(JJ.KK)
40    CONTINUE
          IF(ICHG.EQ.0)GOTO 30
          ZIC = AIMAG(ZI)
          CZCHK = ABS(ZIC)
          IF(CZCHK.EQ.0.)GOTO 208
          IF(INP.EQ.I)GOTO 208
          ZI = CONJG(ZI)
          INP = I
          INULLC = INULLC + I
          ICHG = ICHG + 1
          ESLOW(INULLC) = Z1
          E(ICHG) = Z1
          EO(ICHG) = ZI
          GOTO 34
30    CONTINUE
C
C-------------FIND LEFT/RIGHT RESIDUALS--------------------------------
C
208   COBJR = 0.0
          COBJL = 0.0
          DO 50 J = I.IROWS
          DO 55 I = I.55
          WVR(I) = XSLOW(I.J)
55    WVL(I) = TSLOW(I.J)
C     IF(ITER.NE.0)GOTO 46
```

```
C8    IF(J.LE.4)GOTO 45
C     IF(J.EO.10)GOTO 45
C     IF(J.EO.11)GOTO 45
C     IF(J.EQ.11)GOTO 45
C     IF(J.EQ.21)GOTO 45
C     GOTO 46
C45    CALL SROOT(ESLOW(J),ROT,.0125)
C      WRITE(6,58)J,ROT
C58    FORMAT(1X,'SLOW LEFT EIGENVECTOR NO.',I2,/5X,
C     * 'EIGENVALUE = ',E12.5,1X,E12.5,' J',/5X,
C     *'DESIRED VECTOR      DESIGN VECTOR',/)
C      WRITE(6,56)(TDES(I,J),TSLOW(I,J),I=1,8)
C      WRITE(6,47)
C47    FORMAT('**************************      **************************'
C      WRITE(6,56)(TDES(I,J),TSLOW(I,J),I=9,16)
C      WRITE(6,47)
C      WRITE(6,56)(TDES(I,J),TSLOW(I,J),I=17,55)
C56    FORMAT(IX,E12.5,1X,E12.5,'J',5X,E12.5,1X,E12.5,'J')
46    DO 57 JJ = 1,IUB1
            DO 57 KK = 1,55
57    NMR(JJ,KK) = RNULL(J,JJ,KK)
            DO 60 JJ = 1,N1-IRANCT
            DO 60 KK = 1,N1
60    NML(JJ,KK) = LNULL(J,JJ,KK)
            CALL CMATML(NMR,WVR,IUBI,55,1,ROBJ)
            CALL CMATML(NML,WVL,N1-IRANCT,N1,1,LOBJ)
            RTEST = 0.
            DO 61 JJ = 1,IUB1
            RESR2(J) = RESR2(J) + ROBJ(JJ)*CONJG(ROBJ(JJ))
            RESR1(J) = RESR1(J) + CABS(ROBJ(JJ))
            RESR(J) = RESR(J) + ROBJ(JJ)**2
            RINFR = CABS(ROBJ(JJ))
            IF(RTEST.GT.RINFR)GOTO 61
            REINFR(J) = RINFR
            RTEST = RINFR
61    CONTINUE
            RESR2(J) = SORT(RESR2(J))
            RR = RESR(J)
            RESRR(J) = CSQRT(RR)
C     COBJR = COBJR + CABS(RR)
C     CCSJR = COBJR + CABS(RESRR(J))
            COBJR = CCBJR + RESR2(J)
            LTEST = 0.
            DO 65 JJ = 1,N1-IRANCT
            RESLS(J) = RESLS(J) + LOBJ(JJ)
            RESL2(J) = RESL2(J) + LOBJ(JJ)*CONJG(LOBJ(JJ))
            RESL1(J) = RESL1(J) + CABS(LOBJ(JJ))
            RESL(J) = RESL(J) + LOBJ(JJ)**2
            ELDAM(JJ,J) = LOBJ(JJ)
C-----------------SAVE UNDAMAGED LFT RESIDUAL VECTORS-----------------
            IF(IUND.EO.0)GOTO 63
            ELUND(JJ,J) = LOBJ(JJ)
63    RINFL = CABS(LOBJ(JJ))
            IF(LTEST.GT.RINFL)GOTO 65
            REINFL(J) = RINFL
            LTEST = RINFL
65    CONTINUE
            RESL2(J) = SORT(RESL2(J))
            RL = RESL(J)
            RESLL(J) = CSORT(RL)
C     COBJL = COBJL + CABS(RL)
C     COBJL = COBJL + CABS(RESLL(J))
            COBJL = COBJL + RESL2(J)
50    CONTINUE
C------------------WRITE UNDAMAGED LEFT RESIDUAL VECTOR-----------------
            IF(IUND.EQ.0)GCTO 68
            WRITE(4,3)((ELUND(I,J),J=1,37),I=1,IROWS)
68    IF(IADS.EQ.5)GOTO 93
67    IF(ITER.GT.1)GOTO 400
```

```
C
C--------------DISPLAY RIGHT AND LEFT RESIDUALS-------------------------
66    WRITE(6.70)COBJR.COBJL
70    FORMAT(/5X.'**** DAM-RT SLOW SUBSPACE RESIDUAL = '.E12.5.' ****',
     1 /5X.      '**** DAM-LT SLOW SUBSPACE RESIDUAL = '.E12.5.' ****',/
     2)
C
C-------------DISPLAY INDIVIDUAL RESIDUALS------------------------------
        WRITE(6.80)
80    FORMAT(/1X.'UND EVECT  SLOW STATE   RGT-DAM RESIDUALS          LFT
     1-DAM RESIDUALS'./)
        WRITE(6.81)(ISTATE(I).I.RESR2(I).REINFR(I).RESL2(I).REINFL(I).
     *I=1.IROWS)
81    FORMAT(4X,I2,10X,I2,6X,E12.5,IX,E12.5,2X,E12.5,IX,E12.5)
C
C-------------SET UP ARBITRARY ELEMENTS FOR ADS SIMILAR TO ERSPACE------
C-------------FOR RECR25 DO NOT LET RT STAB OR LT STAB BE ARBITRARY
        III = 1
        DO 71 I = 9.43
C     IF(I.LE.8)GOTO 71
        IARB(III) = I
        III = III + 1
71    CONTINUE
C      COMMENTED OUT FOR RECR25
C     DO 72 I = 29.34
C     IARB(III) = I + 9
C     III = III + 1
C2    CONTINUE
C-------------FOR RECR25 CASE THERE ARE10 LESS ARB ELEMENTS THAN FOR----
C--------------SYMMETRIC CASES. COMMENT THE FOLLOWING TWO OUT FOR SYMM.
C--------------CASES.-----------------------------------------------
C     IARB(27) = 45
C     IARB(28) = 47
C                        USE THE FOLLOWING TWO FOR SYMMETRIC CASES
        IARB(36) = 45
        IARB(37) = 47
C
C-------------LEFT NULL SPACE OPTIMIZATION QUE--------------------------
C
        WRITE(6.82)
82    FORMAT(/5X.'**** ENTER "1" FOR MIN. OF LFT-DAM RESIDUALS ONLY ****
     *'./5X.     '****       USING ARB VECTOR ELEMENTS AS VARIABLES****
     *'./5X.     '**** ENTER "2" FOR LFT NULL SPACE INTRSEC. ANALY. ****
     *'./5X.     '**** ENTER "3" TO SHIFT A DESIRED SLOW EIGENVALUE ****
     *'./5X.     '**** ENTER "4" TO EXIT ELSPACE AND WRITE OPTEIG   ****
     *'./5X.     '**** ENTER "5" FOR MIN. OF LFT-DAM RESIDUALS ONLY ****
     *'./5X.     '****             USING EIGENVALUE AS VARIABLE.     ****
     *'./5X.     '**** ENTER "6" TO WRITE SEGEIG.                    ****
     *'./5X.     '**** ENTER "7" TO EXIT ELSPACE.                    ****
     *'./5X.     '**** ENTER "0" TO EXIT ELSPACE AND WRITE XSLOW.   ****
     *'./)
        READ(*.8)IADS
        IF(IADS.EQ.7)GOTO 9000
        IF(IADS.EQ.6)GOTO 530
        IF(IADS.EQ.4)GOTO 424
        IF(IADS.EQ.1)GOTO 83
        IF(IADS.EQ.5)GOTO 83
        IF(IADS.EQ.2)GOTO 84
        IF(IADS.EQ.0)GOTO 521
        CALL FRTCMS('CLRSCRN ')
C-------------REINIT RESIDUALS-----------------------------------------
        DO 87 II = 1.IROWS
        REINFR(II) = 0.
        REINFL(II) = 0.
        RESR1(II) = 0.
        RESL1(II) = 0.
        RESR2(II) = 0.
        RESL2(II) = 0.
        RESLS(II) = CMPLX(0..0.)
```

```
          RESRS(II) = CMPLX(0.,0.)
          RESR(II) = CMPLX(0.,0.)
87    RESL(II) = CMPLX(0.,0.)
C-------------QUERY USER FOR EIGENVALUE SHIFT INFO---------------------
77    WRITE(6,73)
73    FORMAT(/5X,'**** ENTER UNO VECTOR NO. YOU DESIRE TO SHIFT ****',/5
     1X          ,'**** ENTER "0" TO QUIT.                        ****',/)
          READ(*,2)ICHG
          CALL SROOT(E(ICHG),SCHG,.0125)
78    WRITE(6,74)ICHG,SCHG
74    FORMAT(/5X,'**** ENTER NEW S-PLANE VALUE OF VECTOR ',I2,' ****',/
     *        5X,'     CURRENT VALUE = ',E12.5,1X,E12.5,/,
     *        5X,'                 REAL PART ?',/)
          READ(*,75)RZ
          WRITE(6,79)
79    FORMAT(/5X,'                IMAG PART ?',/)
75    FORMAT(F12.5)
          READ(*,75)CZ
          CALL FRTCMS('CLRSCRN ')
          Z4 = CMPLX(RZ,CZ)
          E(ICHG) = CEXP(Z4*.0125)
          EO(ICHG) = E(ICHG)
          INP = 0
          DO 76 II = 1,IROWS
          IF(ISTATE(II).NE.ICHG)GOTO 76
          ESLOW(II) = E(ICHG)
          INULLC = II
          GOTO 26
76    CONTINUE
          INULLC = II
          GOTO 26
C
C-------------OPTIMIZATION---------------------------------------------
C
83    WRITE(6,105)
105   FORMAT(/5X,'**** ENTER SLOW INDEX OF THE RESIDUAL FOR ADS ****',/)
          READ(*,85)IO(I)
85    FORMAT(I2)
          CALL FRTCMS('CLRSCRN ')
          GOTO 241
84    CALL FRTCMS('CLRSCRN ')
          IF(INULL.NE.0)GOTO 86
          READ(7,2)INULL
          READ(7,3)((RUNULL(1,JJ,KK),KK=1,55),JJ=1,45)
          READ(7,3)((LUNULL(1,JJ,KK),KK=1,55),JJ=1,37)
86    IO(I) = INULL
C
C------------INIT FOR ADS---------------------------------------------
241   IR = IO(1)
          RL = RESL(IR)
          RR = RESR(IR)
          ABR = CABS(RR)
          ABL = CABS(RL)
C     IF(ABR.GT.ABL)GOTO 111
C     OBJ = ABL
C     GOTO 112
111   OBJ = ABL
112   IF(INFO.EQ.1)GOTO 121
113   IN = 1
          ISTRAT = 0
          IOPT = 3
          IONED = 2
          IF(IADS.EQ.1)GOTO 90
          NOV = 2
          GOTO 91
90    NOV = IN*2*55
91    NCON = 0
          NGT = 0
          IGRAD = 0
          NRA = 111
          NCOLA = 222
          IPRINT = 0
C
C-------------SET BOUNDS-------------------------------------------------
C
          Z3 = ESLOW(IR)
          IF(IADS.EQ.1)GOTO 114
          CALL SROOT(Z3,Z2,TSTEP)
          XX(1) = REAL(Z3)
          XX(2) = AIMAG(Z3)
          WRITE(6,95)IR,ESLOW(IR),Z2,ISTATE(IR)
95    FORMAT(/5X,'**** ADS IS BEING ENTERED FOR ESLOW(',I2,') ****',/5X,
     1            '**** Z = ',E12.5,1X,E12.5,'J                ****',/5X,
     2            '**** S = ',E12.5,1X,E12.5,'J                ****',/5X,
     3            '**** ENTER Z-PLANE BOUNDS FOR THIS EIGEN-   ****',/5X,
     4            '**** VALUE IN F20.13 FORMAT.                ****',/5X,
     5            '**** UNDAMAGED EIGENVECTOR NO. ',I2,      '****',/)
                WRITE(6,720)
720     FORMAT(/5X,'**** ENTER VALUE OF ALPHA FOR OBJ ****',/)
                READ(*,117)ALPHA
          GOTO 92
114   CALL SROOT(Z3,Z2,TSTEP)
          WRITE(6,110)IR,ESLOW(IR),Z2,ISTATE(IR)
110   FORMAT(/5X,'**** ADS IS BEING ENTERED FOR ESLOW(',I2,') ****',/5X,
     1            '**** Z = ',E12.5,1X,E12.5,'J                ****',/5X,
     2            '**** S = ',E12.5,1X,E12.5,'J                ****',/5X,
     3            '**** ENTER BOUNDS ON THIS SLOW LFT EVECTOR  ****',/5X,
     4            '**** IN F20.13 FORMAT.                      ****',/5X,
     5            '**** UNDAMAGED EIGENVECTOR NO. ',I2,      '****',/)
92    WRITE(6,115)
115   FORMAT(/5X,'**** REAL TOLERANCE =  ****',/)
          READ(*,117)BR
          WRITE(6,116)
116   FORMAT(/5X,'**** IMAG TOLERANCE =  ****',/)
          READ(*,117)BI
117   FORMAT(F20.13)
C     WRITE(6,118)
C118  FORMAT(/5X,'**** ENTER WEIGHTING FACTOR ****',/)
C     READ(*,117)WEIGHT
          CALL FRTCMS('CLRSCRN ')
C       BUILD DESIGN VARIABLE VECTOR AND UPPER/LOWER BOUND VECTOR
          IF(IADS.EQ.1)GOTO 239
          VUB(1) = XX(1) + BR
          VLB(1) = XX(1) - BR
          VUB(2) = XX(2) + BI
          VLB(2) = XX(2) - BI
          IF(XX(2).NE.0.)GOTO 94
          IIMAG = 1
94    DF(1) = 0.
          DF(2) = 0.
          CALL FRTCMS('CLRSCRN ')
C--------------------------------------
93    Z5 = RESL(IR)
C-----PRE 24 OCT OBJ-------
C     OBJ = CABS(Z5)
C-----24 OCT NEW OBJ
C     Z5A = CSQRT(Z5)
C     OBJ = CABS(Z5)
C     OBJ = RESL1(IR)
          OBJ = RESL2(IR) - ALPHA
C     OBJ = REINFL(IR)
          IPRINT = 0
C------------------2-NORM OF LEFT RESIDUAL DIFFERENCES-----------------
C      DO 700 II = 1,37
C700   DIFF(II) = ELDAM(II,IR) - ELUND(II,IR)
C      OBJ = 0.
C      DO 710 II = 1,37
C710   OBJ = OBJ + DIFF(II)*CONJG(DIFF(II))
```

```
C      OBJ = SQRT(OBJ) - ALPHA
C------------------INFINITY NORM OF LEFT RESIDUAL DIFFERENCES----------
C        OBJ = 0.
C        DTEST = 0.
C        DO 715 II = 1,37
C        DINF = CABS(DIFF(II))
C        IF(DTEST.GT.DINF)GOTO 715
C        OBJ = DINF
C        DTEST = DINF
C715      CONTINUE
            CALL ADS(INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,XX,VLB,VUB,
          1 OBJ,G,IDG,NGT,IC,DF,RA,NRA,NCOLA,WK,10000,IWK,1200)
            IF(IIMAG.NE.1)GOTO 99
            XX(2) = 0.
99     WRITE(6,97)ITER,XX(1),XX(2),OBJ
97     FORMAT(1X,'ITER NO.',I2,' XX(1) = ',E13.6,' XX(2) = ',E13.6
          + ,' OBJ = ',E12.5)
            ITER = ITER + 1
            ICHG = ISTATE(IR)
            Z4 = CMPLX(XX(1),XX(2))
C      E(ICHG) = CEXP(Z4*.0125)
            E(ICHG) = Z4
            ESLOW(IR) = E(ICHG)
            ED(ICHG) = E(ICHG)
            INULLC = IR
            IF(INFO.EQ.0)GOTO 98
            IF(ITER.GT.20)GOTO 98
            IF(OBJ.LT.1.E-01)GOTO 98
            INP = 0
            DO 96 I = 1,IROWS
            RESRS(I) = CMPLX(0.,0.)
            RESLS(I) = CMPLX (0.,0.)
            RESR1(I) = 0.
            RESL1(I) = 0.
            RESR2(I) = 0.
            RESL2(I) = 0.
            REINFR(I) = 0.
            REINFL(I) = 0.
            RESR(I) = CMPLX(0.,0.)
96     RESL(I) = CMPLX(0.,0.)
            GOTO 26
98     IADS = 0
            IR = 0
            ICHG = 0
            ITER = 0
            INFO = 0
            IIMAG = 0
            INP = 0
            GOTO 66
239    JJ = I
            KK = IO(1)
            DO 120 LL = 1,55
            RX = REAL(XSLOW(LL,KK))
            CX = IMAG(XSLOW(LL,KK))
            TR = REAL(TSLOW(LL,KK))
            TC = IMAG(TSLOW(LL,KK))
125    XX(JJ) = TR
            XX(JJ+1) = TC
C------------CURRENTLY  IMAG PART OF THE VECTOR ARE NOT ARB FOR-------
C              RECR25 CASE   25% ASYMMETRIC DAMAGE CASE(NOT FOR SYMM)
C  27 ARBITRARY ELEMENTS FOR RECR25 USE MM = 1,36 FOR SYMMETRIC CASE
C!!!!!!!!!!!!!!!!!!!!!!NOTE!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
            DO 126 MM = 1,37
126    IF(IARB(MM).EQ.LL)GOTO 127
            VUB(JJ) =  XX(JJ) + BR
            VLB(JJ) =  XX(JJ) - BR
            GOTO 128
127    VUB(JJ) = XX(JJ) + BR
            VLB(JJ) = XX(JJ) - BR
```

```
128   IF(TC.NE.0.0)GOTO 286
            VUB(JJ+1) = 0.0
            VLB(JJ+1) = 0.0
            GOTO 285
286   DO 287 MM = 1,37
287   IF(IARB(MM).EQ.LL)GOTO 288
            VUB(JJ+1) = XX(JJ+1) + BI
            VLB(JJ+1) = XX(JJ+1) - BI
            GOTO 285
288   VUB(JJ+1) = XX(JJ+1) + BI
            VLB(JJ+1) = XX(JJ+1) - BI
285   DF(JJ) = 0.
            DF(JJ+1) = 0.
C
C-------------ZEROIZE OUT ELEMENTS WHICH ARE < +/- 1.D-08--------------
C               AM NOT ZEROIZING OUT FOR RECR25 CASE
C     RTEST = ABS(XX(JJ))
C     CTEST = ABS(XX(JJ+1))
C     IF(RTEST.EQ.0.)GOTO 130
C     IF(RTEST.GT.1.E-08)GOTO 130
C     XX(JJ) = 1.E-16
C     VUB(JJ) = 1.E-16
C     VLB(JJ) = -1.0E-16
C130   IF(CTEST.EQ.0.)GOTO 265
C      IF(CTEST.GT.1.E-08)GOTO 265
C      XX(JJ+1) = 1.E-16
C     VUB(JJ+1) = 1.0E-16
C     VLB(JJ+1) = -1.0E-16
265   JJ = JJ + 2
120   CONTINUE
121   CALL ADS(INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,XX,VLB,VUB,
         1 OBJ,G,IDG,NGT,IC,DF,RA,NRA,NCOLA,WK,10000,IWK,1200)
            ITER = ITER + 1
C        EVALUATE OBJECTIVE FUNCTION
            JJ = 1
            ICONJG = 0
            KK = IO(1)
            DO 245 LL = 1,55
            IF(IMAG(ESLOW(KK)).EQ.0.0)GOTO 246
271   ICONJG = 1
246   TSLOW(LL,KK) = CMPLX(XX(JJ),XX(JJ+1))
248   JJ = JJ + 2
245   CONTINUE
            OBJ = 0.
            RLREAL = 0.
            RL = CMPLX(0.,0.)
            RESL1(IR) = 0.0
            RESL2(IR) = 0.0
            DO 290 II = 1,55
290   WVL(II) = TSLOW(II,IR)
            DO 291 JJ = 1,N1-IRANCT
            DO 291 KK = 1,N1
291   NML(JJ,KK) = LNULL(IR,JJ,KK)
            CALL CMATML(NML,WVL,N1-IRANCT,N1,1,LOBJ)
            LTEST = 0.
            DO 292 JJ = 1,N1-IRANCT
C292   RLREAL = RLREAL + CABS(LOBJ(JJ))
C-------PLAN 3 OBJ= SUM OF RL'S--------
            RL = RL + LOBJ(JJ)
C-------ASYMM TEST OBJ = 1 NORM------
            RESL1(IR) = RESL1(IR) + CABS(LOBJ(JJ))
            RESL2(IR) = RESL2(IR) + LOBJ(JJ)*CONJG(LOBJ(JJ))
C-------ASYMM CASE TEST, OBJ = INFINITY NORM------
            RINFL = CABS(LOBJ(JJ))
            IF(LTEST.GT.RINFL)GOTO 292
            REINFL(IR) = RINFL
            LTEST = RINFL
292   CONTINUE
            RESL2(IR) = SQRT(RESL2(IR))
```

```
C-------PLAN 1 OBJ----------
C292    RL = RL + LOBJ(JJ)**2
C*******************************************
              IF(IADS.NE.2)GOTO 294
C*******************************************
              RLD = CABS(RL)
              RL = CMPLX(0.,0.)
              DO 293 JJ = 1,37
              DO 293 KK = 1,N1
293     NML(JJ,KK) = LUNULL(1,JJ,KK)
              CALL CMATML(NML,WVL,37,N1,1,LCBJ)
              DO 295 JJ = 1,37
295     RL = RL + LOBJ(JJ)
              RLU = CABS(RL)
              IF(RLU.GT.RLD)GOTO 298
              OBJ = RLD + WEIGHT*RLU
              GOTO 296
298     OBJ = RLU + WEIGHT*RLD
              GOTO 296
C294    OBJ = RLREAL
C-------PLAN 3 OBJ----------
C294    OBJ = CABS(RL)
C-------PLAN 1 CBJ----------
C294    OBJC = CSCRT(RL)
C       OBJ = CABS(OBJC)
C--------------INF-NORM OBJECTIVE FOR ASYMMETRIC CASE 11/3/86-----
C      OBJ = REINFL(IR)
C--------------2-NORM OBJECTIVE FUNCTION 11/3/86-----------------
294    OBJ = RESL2(IR)
C--------------2-NORM + ORIENTATION OBJECTIVE FUNCTION 11/6/86---------
                     DO 800 JJ = 1,55
800       DT(JJ) = TDES(JJ,IR) - TSLOW(JJ,IR)
                     DTNORM(IR) = 0.
                     DO 810 JJ = 1,55
810       DTNORM(IR) = DTNORM(IR) + DT(JJ)*CONJG(DT(JJ))
                     DTNORM(IR) = SORT(DTNORM(IR))
C       OBJ = DTNORM(IR) + RESL2(IR)
296    IF(ITER.GT.2000)GOTO 257
              IF(INFO.EO.0)GCTO 257
              GOTO I21
257    IF(ICONJG.EQ.0)GOTO 249
              KK = IO(1)
              DO 305 LL = 1,55
305    TSLOW(LL,KK+1) = CONJG(TSLOW(LL,KK))
C
C------------REBUILD X,T,XLOW,TSLOW-------------------------------------
C
249    DO 300 KK = 1,55
              DO 302 LL = I,IROWS
              IF(ISTATE(LL).EQ.KK)GOTO 301
302    CONTINUE
              DO 303 JJ = I,55
              XS(JJ,KK) = T(JJ,KK)
303    XT(JJ,KK) = XS(JJ,KK)
              GOTO 300
301    DO 304 JJ = 1,55
              XS(JJ,KK) = TSLOW(JJ,LL)
304    XT(JJ,KK) = XS(JJ,KK)
300    CONTINUE
              CALL LEQT1C(XS,55,55,UNITYC,55,55,0,WA,IER)
              DO 247 KK = I,55
              DO 247 LL = I,55
              T(KK,LL) = XT(KK,LL)
              XD(LL,KK) = UNITYC(KK,LL)
247    X(LL,KK) = UNITYC(KK,LL)
              DO 255 JJ = 1,IROWS
              DO 255 KK = 1,55
              LL = ISTATE(JJ)
255    XSLOW(KK,JJ) = X(KK,LL)
```

```
         DO 250 KK = 1,55
         DO 250 LL = 1,55
250   UNITYC(KK,LL) = CMPLX(0.0,0.0)
         DO 252 KK = 1,55
252   UNITYC(KK,KK) = CMPLX(1.0,0.0)
         DO 475 I = 1,IROWS
         RESR1(I) = 0.
         RESL1(I) = 0.
         RESR2(I) = 0.
         RESL2(I) = 0.
         RESR(I) = CMPLX(0.,0.)
475   RESL(I) = CMPLX(0.,0.)
         GOTO 208
400   CALL FRTCMS('CLRSCRN ')
C
C--------------DISPLAY OBJECTIVE FUNCTION-------------------------------
C
         WRITE(6,410)OBJ,RESL2(IR),DTNORM(IR),INFO,ITER,RLD,RLU,WEIGHT
410   FORMAT(/5X,'****        ADS COMPLETE          ****',/5X,
     *            '    OBJECTIVE FUNCTION = ',E12.5,/5X,
     *            '    2-NORM OF RESIDUAL = ',E12.5,/5X,
     *            '    DT 2-NORM OF DIFF  = ',E12.5,/5X,
     *            '    INFO = ',I1,'    ITER = ',I4,/5X,
     *            ' DAM-LT OBJ = ',E12.5,1X,' UND-LT OBJ = ',E12.5,/5X,
     *            ' WEIGHT FACTOR = ',E12.5,/)
C
C-------------DECIDE TO REENTER ADS, WRITE ADSDAT FOR ELSPACE RENTRY OR-
C            WRITE OPTEIG FOR RECONF RENTRY.
C
         WRITE(6,470)
470   FORMAT(/5X,'**** ENTER "1" TO REENTER ADS    ****',/5X,
     *            '**** ENTER "2" TO WRITE ADSDATA ****',/5X,
     *            '**** ENTER "3" TO WRITE XSLOW    ****',/5X,
     *            '**** ENTER "4" TO WRITE SEGEIG   ****',/5X,
     *            '**** ENTER "0" TO WRITE OPTEIG   ****',/)
         READ(*,8)IEEN
         CALL FRTCMS('CLRSCRN ')
         IF(IEEN.EQ.0)GOTO 424
         IF(IEEN.EQ.2)GOTO 510
         IF(IEEN.EQ.3)GOTO 521
         IF(IEEN.EQ.4)GOTO 530
         DO 500 I = 1,24
         RESR1(I) = 0.
         RESL1(I) = 0.
         RESR2(I) = 0.
         RESL2(I) = 0.
         REINFR(I) = 0.
         REINFL(I) = 0.
         RESR(I) = CMPLX(0.,0.)
500   RESL(I) = CMPLX(0.,0.)
         INFO = 0
         ITER = 0
         GOTO 208
C
C-------------WRITE FILE 08 OPTEIG FOR USE BY RECONF-------------------
C
C--------INSERT  ACT+  SENSOR          UND ELEMENTS------------
C--------VALUES INTO SLOW EIGENVECTORS--WEO/LAT ELE 9-16----------------
C424    DO 462 I = 5,43
C424    DO 462 I = 9,43
C       DO 462 J = 38,55
C      IF(J.NE.41)GOTO 467
C      IF(J.NE.42)GOTO 467
C      IF(J.NE.45)GOTO 467
C      IF(J.NE.52)GOTO 467
C      IF(I.NE.9)GOTO 467
C      IF(I.NE.10)GOTO 467
C      IF(I.NE.11)GOTO 467
C      IF(I.NE.12)GOTO 467
```

```
C      IF(I.NE.13)GOTO 467
C      IF(I.NE.14)GOTO 467
C      IF(I.NE.15)GOTO 467
C      IF(I.NE.16)GOTO 467
C      GOTO 462
C      XD(I,J) = XSAVE(I,J)
C462   CONTINUE
C--------INSERT UND LAT ELEMENTS INTO SLOW LONG EIGENVECTORS-----------
C      DO 464 I = 5,8
C      XD(I,38) = XSAVE(I,38)
C      XD(I,39) = XSAVE(I,39)
C      XD(I,43) = XSAVE(I,43)
C64    XD(I,46) = XSAVE(I,46)
C--------INSERT UND LON ELEMENTS INTO SLOW LAT EIGENVECTORS------------
C      DO 466 I = 1,4
C      XD(I,41) = XSAVE(I,41)
C      XD(I,42) = XSAVE(I,42)
C      XD(I,45) = XSAVE(I,45)
C66    XD(I,52) = XSAVE(I,52)
C        -RECONSTRUCT FAST EIGENSPACE FROM UNDAMAGED EIGENSPACE- RECR25--
C        -FOR RECR25 7,8,17,18,23,24 ARE ALSO SLOW STATES-
C424    DO 463 I = 9,43
C       DO 463 J = 1,37
C      IF(J.EQ.7)GOTO 463
C      IF(J.EQ.8)GOTO 463
C      IF(J.EQ.17)GOTO 463
C      IF(J.EQ.18)GOTO 463
C      IF(J.EQ.23)GOTO 463
C      IF(J.EQ.24)GOTO 463
C        XD(I,J) = XSAVE(I,J)
C463    CONTINUE
C-------------INSERT FAST ELEMENTS OF CONTROL LAW FILTERS-------------
C              DO 465 J = 1,37
C              XD(45,J) = XSAVE(45,J)
C465            XD(47,J) = XSAVE(47,J)
C-------------RESTRUCTURE X,DELTA BY REORDERING THE FIRST 18------------
C              VECTOR/EIGENVALUE PAIRS TO BE THE SLOW EIGENSPACE
424       DO 900 I = 1,55
                  DO 900 J = 1,18
900       XDR(I,J) = XD(I,J+37)
                  DO 910 I = 1,55
                  DO 910 J = 19,55
910       XDR(I,J) = XD(I,J-18)
                  DO 920 I = 1,18
920       EDR(I) = ED(I+37)
                  DO 925 I = 19,55
925       EDR(I) = ED(I-18)
                  DO 930 I = 1,55
930       ED(I) = EDR(I)
                  DO 935 I = 1,55
                  DO 935 J = 1,55
935       XD(I,J) = XDR(I,J)
C--------------REINSET FAST EIGENSPACE--------------------------------
C        DO 936 I = 9,43
C        DO 936 J = 19,55
C        XD(I,J) = XSAVE(I,J)
C936    CONTINUE
C-------------QUERY USER TO CHECK RECONSTRUCTED OPT SPACE FOR RESIDUALS-
              WRITE(6,610)
610    FORMAT(/5X,'*ENTER "1" TO CALC RESIDS FOR RECONSTRUCTED SPACE*',
     *        /5X,'*ELSE "0".                                    *',/)
            READ(*,8)IRECHK
            IF(IRECHK.EQ.0)GOTO 468
            DO 630 I = 1,55
            DO 630 J = 1,55
            DR = DREAL(XD(I,J))
            DC = DIMAG(XD(I,J))
            R = SNGL(DR)
            C = SNGL(DC)
```

```
630   XS(I,J) = CMPLX(R,C)
            DO 645 I = 1,55
            DO 645 J = 1,IROWS
            KK = ISTATE(J)
645   XSLOW(I,J) = XS(I,KK)
            DO 650 I = 1,24
            RESL1(I) = 0.
            RESR1(I) = 0.
            RESL2(I) = 0.
            RESR2(I) = 0.
            REINFR(I) = 0.
            REINFL(I) = 0.
            RESR(I) = CMPLX(0.,0.)
650   RESL(I) = CMPLX(0.,0.)
            INFO = 0
            ITER = 0
            GOTO 620
468   WRITE(3,460)((XO(I,J),J=1,55),I=1,55)
            WRITE(8,460)(ED(I),I=1,55)
460   FORMAT(4D20.13)
            GOTO 9000
C
C-------------WRITE FILE IPASS XSLOW FOR DECOMP------------------------
C
521   IPASS = IPASS + 1
            WRITE(IPASS,2)IROWS,IPASS
            WRITE(IPASS,4)(ISTATE(I),I=1,IROWS)
            WRITE(IPASS,3)((X(I,J),J=1,55),I=1,55)
            WRITE(IPASS,3)((T(I,J),J=1,55),I=1,55)
            WRITE(IPASS,3)((XSLOW(I,J),J=1,IROWS),I=1,55)
            WRITE(IPASS,3)((TSLOW(I,J),J=1,IROWS),I=1,55)
            WRITE(IPASS,3)(ESLOW(I),I=1,IROWS)
            GOTO 9000
C
C-------------WRITE FILE IPASS ADSDAT FOR USE BY ELSPACE------------
C
510   IPASS = IPASS + 1
            WRITE(IPASS,2)IROWS,IPASS
            WRITE(IPASS,4)(ISTATE(I),I=1,IROWS)
            WRITE(IPASS,3)((X(I,J),J=1,55),I=1,55)
            WRITE(IPASS,3)((T(I,J),J=1,55),I=1,55)
            WRITE(IPASS,3)((XSLOW(I,J),J=1,IROWS),I=1,55)
            WRITE(IPASS,3)((TSLOW(I,J),J=1,IROWS),I=1,55)
            WRITE(IPASS,3)(ESLOW(I),I=1,IROWS)
            GOTO 9000
C
C-------------WRITE FILE 9,10,OR 11 SEGEIG DATA------------------------
C
530   CALL FRTCMS('CLRSCRN ')
            WRITE(6,535)
535   FORMAT(/5X,'**** ENTER DESIGN SEGMENT NUMBER ****',/)
            READ(*,8)ISEG
            ISEG = ISEG + 8
            WRITE(ISEG,3)((XSLOW(I,J),J=1,IROWS),I=1,55)
            WRITE(ISEG,3)(ESLOW(I),I=1,IROWS)
9000  STOP
            END
C**********************************************************************
C
C                 SUBROUTINE CMATML (COMPLEX MATRIX MULTIPLICATION)
C
C        COMPUTES: YY = AA * BB
C        IA = # OF ROWS IN AA
C        LL = # OF ROWS IN BB AND # OF COLUMNS IN AA
C        IB = # OF COLUMNS IN BB
C**********************************************************************
            SUBROUTINE CMATML(AA,BB,IA,LL,IB,YY)
            COMPLEX*8 AA(55,55),BB(55,55),YY(55,55)
            INTEGER IA,LL,IB
            DO 30 I = 1,IA
            DO 20 J = 1,IB
            YY(I,J) = CMPLX(0.,0.)
            DO 10 INDEX = 1,LL
            YY(I,J) = YY(I,J) + AA(I,INDEX) * BB(INDEX,J)
10    CONTINUE
20    CONTINUE
30    CONTINUE
            RETURN
            END
C***********************************************************************
C
C             SUBROUTINE SROOT: CONVERTS Z-PLANE ROOTS TO
C                               S-PLANE ROOTS
C
C                   CDR V.F. GAVITO   JULY 1986
C
C***********************************************************************
            SUBROUTINE SROOT(Z,S,T)
            IMPLICIT REAL*4(A-H,P-Z)
            COMPLEX*8 Z,S
            RS = IMAG(Z)
            RC = REAL(Z)
            B = (ATAN2(RS,RC))/T
            RZ = CABS(Z)
            A = -LOG(RZ)/T
            S = CMPLX(-A,B)
            RETURN
            END
```

APPENDIX G

SAMPLE RECONFIGURATION RUN FOR F-18, CASE A DAMAGE

```
RECONF
VS FORTRAN COMPILER ENTERED.  12:59:29

**MAIN** END OF COMPILATION 1 ******

**CMATML** END OF COMPILATION 2 ******

**RANKD** END OF COMPILATION 3 ******

**WRMATD** END OF COMPILATION 4 ******

**SRCOT** END OF COMPILATION 5 ******
VS FORTRAN COMPILER EXITED.   12:59:36


LOADING RECONF AND ORACLSA
EXECUTION BEGINS...

        ENTER "1" TO WRITE UNDAMAGED EIGENSTRUCTURE,&
                            SYST. MATRICES.(READ "RECONF" DATA A1,
                            "UNDIEG"."UNDABC".& "CLM".
        ENTER "2" TO READ UNDAMAGED  EIGENSTRUCTURE AND
                            WRITE SLOW RT SUBPACE DATA FOR USE BY
                            "ERSPACE" (WRITE "ERSPACE" DATA A1)
        ENTER "3" TO COMPUTE RECONFIGURED GAINS.
                            (WRITE "FRECON" DATA A1)
        ENTER "4" TO COMPUTE RECONFIG. EIGENSTRUCTURE
                            AND TIME RESPONSE (OPTMATD,OPTPLOT)
        ENTER "5" TO COMPUTE Q AND WRITE FILE 12
                            "FXACT" DATA.

2

        **** ENTER "1" TO MODIFY DESIRED EIGENSTRUCTURE ****
        ****             ELSE ENTER "0".****

0

        **** ENTER "1" TO DISPLAY DESIRED EIGENSTRCTURE****
        **** ENTER "0" OTHERWISE.                      ****

0

        **** CALLING LSVDF FOR B ****


        **** LSVDF OF "B" COMPLETE ****

          **** RANK(B) = 10         ****

        **** CALLING LSVDF FOR C ****


        **** LSVDF OF "C" COMPLETE ****

        **** RANK(C) = 18       ****
SLOW STATE NO.  1 UND STATE NO. 38 S = -0.22794E+01  0.27016E+01J
SLOW STATE NO.  2 UND STATE NO. 39 S = -0.22794E+01 -0.27016E+01J
SLOW STATE NO.  3 UND STATE NO. 40 S = -0.38114E+01  0.00000E+00J
SLOW STATE NO.  4 UND STATE NO. 41 S = -0.18703E+01  0.26152E+01J
SLOW STATE NO.  5 UND STATE NO. 42 S = -0.18703E+01 -0.26152E+01J
SLOW STATE NO.  6 UND STATE NO. 43 S = -0.51146E+00  0.00000E+00J
SLOW STATE NO.  7 UND STATE NO. 44 S = -0.11588E+01  0.00000E+00J
SLOW STATE NO.  8 UND STATE NO. 45 S = -0.29042E+01  0.00000E+00J
SLOW STATE NO.  9 UND STATE NO. 46 S = -0.17330E-01  0.00000E+00J
SLOW STATE NO. 10 UND STATE NO. 47 S = -0.20036E+01  0.00000E+00J
SLOW STATE NO. 11 UND STATE NO. 48 S = -0.14434E+01  0.00000E+00J
SLOW STATE NO. 12 UND STATE NO. 49 S =  0.36383E-02  0.00000E+00J
SLOW STATE NO. 13 UND STATE NO. 50 S = -0.10000E+01  0.00000E+00J
SLOW STATE NO. 14 UND STATE NO. 51 S =  0.00000E+00  0.00000E+00J
SLOW STATE NO. 15 UND STATE NO. 52 S =  0.29118E-02  0.00000E+00J
SLOW STATE NO. 16 UND STATE NO. 53 S =  0.00000E+00  0.00000E+00J
SLOW STATE NO. 17 UND STATE NO. 54 S = -0.20001E+01  0.00000E+00J
SLOW STATE NO. 18 UND STATE NO. 55 S = -0.20001E+01  0.00000E+00J
R; T=9.39/11.23 13:00:04
ERSPACE
VS FORTRAN COMPILER ENTERED.  13:02:32

**MAIN** END OF COMPILATION 1 ******

**CMATML** END OF COMPILATION 2 ******

**SROOT** END OF COMPILATION 3 ******
VS FORTRAN COMPILER EXITED.   13:02:35


LOADING ERSPACE
EXECUTION BEGINS...

        **** INDICES OF SLOW EIGENVALUES ****

  38    39    40    41    42

  43    44    45    46    47

  48    49    50    51    52

  53    54    55


        **** THE FOLLOWING VECTOR DISPLAYS THE NEARNESS OF  ****
        **** EACH SLOW EIGENVECTOR TO THE ALLOWABLE RIGHT    ****
        **** HAND SLOW EIGENSPACE FOR THE DAMAGED F18 MODEL. ****

    0.349199582017 8D-02
    0.3491995820178D-02
    0.2956543499654D-02
    0.4303230273294D-02
    0.4303230273294D-02
    0.1927162332338D-02
    0.2176680703960D-02
    0.4067147444816D-02
    0.2923381215272D-03
    0.7099626971567D-04
    0.1498364394931D-02
    0.7207041731954D-02
    0.2856438898831D-11
    0.2550741225155D-12
    0.8990287299080D-05
    0.8946468211520D-10
    0.4907175134379D-12
    0.0000000000000D+00

        **** ENTER "1" IF RESIDUAL IS UNSAT TO RENTER ADS ****
        **** ELSE ENTER "0".                             ****

 0

        **** WRITING ELSPACE DATA FILE 08 ****
```

```
R; T=08.11/09.92 13:03:02
ELSPACE
VS FORTRAN COMPILER ENTERED.  13:04:19

**MAIN** END OF COMPILATION 1 ******

**CMATML** END OF COMPILATION 2 ******

**SROOT** END OF COMPILATION 3 ******
VS FORTRAN COMPILER EXITED.   13:04:25


LOADING ELSPACE ACS
EXECUTION BEGINS...

        **** RANK(C**T) = 18 ****


        **** ENTER "1" IF PASS IS FOR UNDAM ANALYSIS ****
        **** ENTER "0" ELSE.                          ****

0

        **** ENTER "1" IF FIRST PASS THRU ELSPACE ****
        **** ENTER PASS NUMBER OTHERWISE.         ****

1

        **** DAM-RT SLOW SUBSPACE RESIDUAL =  0.36547E-01 ****
        **** DAM-LT SLOW SUBSPACE RESIDUAL =  0.10511E+02 ****


UND EVECT  SLOW STATE   RGT-DAM RESIDUALS           LFT-DAM RESIDUALS

     38          1       0.34926E-02  0.31823E-02   0.33484E+02  0.23673E+02
     39          2       0.34926E-02  0.31823E-02   0.33484E+02  0.23673E-02
     40          3       0.29547E-02  0.26955E-02   0.37790E+02  0.26721E+02
     41          4       0.43344E-02  0.36901E-02   0.90522E-02  0.63941E-02
     42          5       0.43344E-02  0.36901E-02   0.90522E-02  0.63941E-02
     43          6       0.19289E-02  0.17545E-02   0.10999E+02  0.77760E+01
     44          7       0.21797E-02  0.19839E-02   0.68938E-01  0.48743E-01
     45          8       0.42476E-02  0.37847E-02   0.10488E+00  0.74161E-01
     46          9       0.29235E-03  0.26580E-03   0.15057E+01  0.12600E+01
     47         10       0.68759E-04  0.50844E-04   0.20671E-01  0.14616E-01
     48         11       0.14587E-02  0.13553E-02   0.11536E-01  0.81495E-02
     49         12       0.72043E-02  0.65518E-02   0.69335E+00  0.69318E+00
     50         13       0.29570E-11  0.20878E-11   0.27217E-01  0.19516E-01
     51         14       0.25527E-12  0.17498E-12   0.20157E-01  0.20054E-01
     52         15       0.45755E-03  0.45448E-03   0.51955E-01  0.36294E-01
     53         16       0.89445E-10  0.78303E-10   0.00000E+00  0.00000E+00
     54         17       0.49205E-12  0.38527E-12   0.00000E+00  0.00000E+00
     55         18       0.00000E+00  0.00000E+00   0.00000E+00  0.00000E+00

        **** ENTER "1" FOR MIN. OF LFT-DAM RESIDUALS ONLY ****
        ****       USING ARB VECTOR ELEMENTS AS VARIABLES****
        **** ENTER "2" FOR LFT NULL SPACE INTRSEC. ANALY. ****
        **** ENTER "3" TO SHIFT A DESIRED SLOW EIGENVALUE ****
        **** ENTER "4" TO EXIT ELSPACE AND WRITE OPTEIG   ****
        **** ENTER "5" FOR MIN. OF LFT-DAM RESIDUALS ONLY ****
        ****            USING EIGENVALUE AS VARIABLE.     ****
        **** ENTER "6" TO WRITE CEGEIG.                   ****
        **** ENTER "7" TO EXIT ELSPACE.                   ****
        **** ENTER "0" TO EXIT ELSPACE AND WRITE XSLOW.   ****

1

        **** ENTER SLOW INDEX OF THE RESIDUAL FOR ACS ****

01
```

```
        **** ADS IS BEING ENTERED FOR ESLOW( 1) ****
        **** Z =  0.97136E+00  0.32815E-01J                **** 
        **** S = -0.22790E+01  0.27016E+01J                ****
        **** ENTER BOUNDS ON THIS SLOW LFT EVECTOR  ****
        **** IN F20.13 FORMAT.                      ****
        **** UNDAMAGED EIGENVECTOR NO. 38****


        **** REAL TOLERANCE =  ****

0.001

        **** IMAG TOLERANCE =  ****

0.001

        ****       ADS COMPLETE        ****
               OBJECTIVE FUNCTION =  0.61855E+01
               2-NORM OF RESIDUAL =  0.61855E+01
               DT 2-NORM OF DIFF  =  0.27028E-02
               INFO = 0   ITER =  535
          DAM-LT OBJ =  0.00000E+00  UND-LT OBJ =  0.00000E+00
          WEIGHT FACTOR =  0.10000E-01


        **** ENTER "1" TO REENTER ADS   ****
        **** ENTER "2" TO WRITE ADSDATA ****
        **** ENTER "3" TO WRITE XSLOW   ****
        **** ENTER "4" TO WRITE SEGEIG  ****
        **** ENTER "0" TO WRITE OPTEIG  ****

1

        **** DAM-RT SLOW SUBSPACE RESIDUAL =  0.42648E-01 ****
        **** DAM-LT SLOW SUBSPACE RESIDUAL =  0.70510E+02 ****


UND EVECT  SLOW STATE   RGT-DAM RESIDUALS          LFT-DAM RESIDUALS

      38          1        0.40492E-02  0.37709E-02   0.61855E+01  0.46814E+01
      39          2        0.40488E-02  0.37707E-02   0.61855E+01  0.46814E+01
      40          3        0.32182E-02  0.29602E-02   0.37790E+02  0.26721E+02
      41          4        0.46098E-02  0.37388E-02   0.90522E-02  0.63941E-02
      42          5        0.45815E-02  0.37442E-02   0.90522E-02  0.63941E-02
      43          6        0.24062E-02  0.22697E-02   0.10999E+02  0.77762E+01
      44          7        0.26422E-02  0.24819E-02   0.68938E+01  0.48743E+01
      45          8        0.42096E-02  0.36473E-02   0.10488E+00  0.74161E-01
      46          9        0.37225E-03  0.35227E-03   0.15057E+01  0.12600E+01
      47         10        0.11849E-03  0.94354E-04   0.20671E-01  0.14616E-01
      48         11        0.15197E-02  0.13792E-02   0.11536E-01  0.81495E-02
      49         12        0.91837E-02  0.86928E-02   0.69335E+00  0.69318E+00
      50         13        0.53323E-04  0.47874E-04   0.27317E-01  0.19316E-01
      51         14        0.37652E-05  0.36575E-05   0.23157E-01  0.23054E-01
      52         15        0.42540E-03  0.42198E-03   0.51955E-01  0.36294E-01
      53         16        0.93229E-03  0.67544E-03   0.00000E+00  0.00000E+00
      54         17        0.27015E-03  0.26852E-03   0.00000E+00  0.00000E+00
      55         18        0.34582E-05  0.34342E-05   0.00000E+00  0.00000E+00

        **** ENTER "1" FOR MIN. OF LFT-DAM RESIDUALS ONLY ****
        ****        USING ARB VECTOR ELEMENTS AS VARIABLES****
        **** ENTER "2" FOR LFT NULL SPACE INTRSEC. ANALY. ****
        **** ENTER "3" TO SHIFT A DESIRED SLOW EIGENVALUE ****
        **** ENTER "4" TO EXIT ELSPACE AND WRITE OPTEIG   ****
        **** ENTER "5" FOR MIN. OF LFT-DAM RESIDUALS ONLY ****
        ****            USING EIGENVALUE AS VARIABLE.     ****
        **** ENTER "6" TO WRITE SEGEIG.                   ****
        **** ENTER "7" TO EXIT ELSPACE.                   ****
        **** ENTER "0" TO EXIT ELSPACE AND WRITE XSLOW.   ****
```

```
1

        **** ENTER SLOW INDEX OF THE RESIDUAL FOR ADS ****

03

        **** ADS IS BEING ENTERED FOR ESLOW( 3) ****
        **** Z =  0.95347E+00  0.00000E+00J                 ****
        **** S = -0.38118E+01  0.00000E+00J                 ****
        **** ENTER BOUNDS ON THIS SLOW LFT EVECTOR  ****
        **** IN F20.13 FORMAT.                      ****
        **** UNDAMAGED EIGENVECTOR NO. 40****


        **** REAL TOLERANCE =  ****

0.001

        **** IMAG TOLERANCE =  ****

0.001

        ****       ADS COMPLETE        ****
              OBJECTIVE FUNCTION =  0.86096E+01
              2-NORM OF RESIDUAL =  0.86096E+01
              DT 2-NORM OF DIFF  =  0.28879E-02
              INFO = 0    ITER =  536
          DAM-LT OBJ =  0.00000E+00  UND-LT OBJ =  0.00000E+00
          WEIGHT FACTOR =  0.10000E-01


        **** ENTER "1" TO REENTER ADS    ****
        **** ENTER "2" TO WRITE ADSDATA ****
        **** ENTER "3" TO WRITE XSLOW    ****
        **** ENTER "4" TO WRITE SEGEIG   ****
        **** ENTER "0" TO WRITE OPTEIG   ****

1

        **** DAM-RT SLOW SUBSPACE RESIDUAL =  0.43721E-01 ****
        **** DAM-LT SLOW SUBSPACE RESIDUAL =  0.41330E+02 ****


UND EVECT  SLOW STATE   RGT-DAM RESIDUALS           LFT-DAM RESIDUALS

      38          1       0.41267E-02  0.38624E-02   0.61855E+01  0.46814E+01
      39          2       0.41385E-02  0.38747E-02   0.61855E+01  0.46814E+01
      40          3       0.31300E-02  0.28791E-02   0.86096E+01  0.54260E+01
      41          4       0.45921E-02  0.37393E-02   0.90522E-02  0.63941E-02
      42          5       0.46600E-02  0.37442E-02   0.90522E-02  0.63941E-02
      43          6       0.25484E-02  0.24186E-02   0.10999E+02  0.77762E+01
      44          7       0.27548E-02  0.26023E-02   0.68938E+01  0.48743E+01
      45          8       0.42097E-02  0.36473E-02   0.10488E+00  0.74161E-01
      46          9       0.39882E-03  0.37983E-03   0.15057E+01  0.12600E+01
      47         10       0.11463E-03  0.89720E-04   0.20671E-01  0.14616E-01
      48         11       0.15328E-02  0.13802E-02   0.11536E-01  0.81495E-02
      49         12       0.98442E-02  0.93774E-02   0.69335E+00  0.69318E+00
      50         13       0.52751E-04  0.47870E-04   0.27317E-01  0.19316E-01
      51         14       0.34719E-05  0.33888E-05   0.23157E-01  0.23054E-01
      52         15       0.42625E-03  0.42294E-03   0.51955E-01  0.36294E-01
      53         16       0.92305E-03  0.66797E-03   0.00000E+00  0.00000E+00
      54         17       0.26203E-03  0.26049E-03   0.00000E+00  0.00000E+00
      55         18       0.31985E-05  0.31772E-05   0.00000E+00  0.00000E+00

        **** ENTER "1" FOR MIN. OF LFT-DAM RESIDUALS ONLY ****
        ****        USING ARB VECTOR ELEMENTS AS VARIABLES****
        **** ENTER "2" FOR LFT NULL SPACE INTRSEC. ANALY. ****
        **** ENTER "3" TO SHIFT A DESIRED SLOW EIGENVALUE ****
        **** ENTER "4" TO EXIT ELSPACE AND WRITE OPTEIG   ****
        **** ENTER "5" FOR MIN. OF LFT-DAM RESIDUALS ONLY ****
        ****           USING EIGENVALUE AS VARIABLE.      ****
        **** ENTER "6" TO WRITE SEGEIG.                   ****
        **** ENTER "7" TO EXIT ELSPACE.                   ****
        **** ENTER "0" TO EXIT ELSPACE AND WRITE XSLOW.   ****

1

        **** ENTER SLOW INDEX OF THE RESIDUAL FOR ADS ****

06

        **** ADS IS BEING ENTERED FOR ESLOW( 6) ****
        **** Z =  0.99363E+00  0.00000E+00J                 ****
        **** S = -0.51123E+00  0.00000E+00J                 ****
        **** ENTER BOUNDS ON THIS SLOW LFT EVECTOR  ****
        **** IN F20.13 FORMAT.                      ****
        **** UNDAMAGED EIGENVECTOR NO. 43****


        **** REAL TOLERANCE =  ****

0.001

        **** IMAG TOLERANCE =  ****

0.001

        ****        ADS COMPLETE         ****
                OBJECTIVE FUNCTION =  0.98729E+01
                2-NORM OF RESIDUAL =  0.98729E+01
                OT 2-NORM OF DIFF  =  0.18694E-02
                INFO = 0   ITER =  374
          DAM-LT OBJ =  0.00000E+00  UND-LT OBJ =  0.00000E+00
          WEIGHT FACTOR =  0.10000E-01


        **** ENTER "1" TO REENTER ADS   ****
        **** ENTER "2" TO WRITE ADSDATA ****
        **** ENTER "3" TO WRITE XSLOW   ****
        **** ENTER "4" TO WRITE SEGEIG  ****
        **** ENTER "0" TO WRITE OPTEIG  ****

1

        **** DAM-RT SLOW SUBSPACE RESIDUAL =  0.43688E-01 ****
        **** DAM-LT SLOW SUBSPACE RESIDUAL =  0.40204E+02 ****


UND EVECT  SLOW STATE   RGT-DAM RESIDUALS           LFT-DAM RESIDUALS

     38          1       0.41283E-02  0.38645E-02   0.61855E+01  0.46814E+01
     39          2       0.41369E-02  0.38727E-02   0.61855E+01  0.46814E+01
     40          3       0.31302E-02  0.28793E-02   0.86096E+01  0.54260E+01
     41          4       0.45925E-02  0.37393E-02   0.90522E-02  0.63941E-02
     42          5       0.46608E-02  0.37442E-02   0.90522E-02  0.63941E-02
     43          6       0.25487E-02  0.24189E-02   0.98729E+01  0.69282E+01
     44          7       0.27545E-02  0.26020E-02   0.68938E+01  0.48743E+01
     45          8       0.41914E-02  0.36321E-02   0.10488E+00  0.74161E-01
     46          9       0.39881E-03  0.37981E-03   0.15057E+01  0.12600E+01
     47         10       0.11464E-03  0.89769E-04   0.20671E-01  0.14616E-01
     48         11       0.15327E-02  0.13811E-02   0.11536E-01  0.81495E-02
     49         12       0.98435E-02  0.93766E-02   0.69335E+00  0.69318E+00
     50         13       0.52782E-04  0.47872E-04   0.27317E-01  0.19316E-01
     51         14       0.35469E-05  0.34311E-05   0.23157E-01  0.23054E-01
     52         15       0.42531E-03  0.42198E-03   0.51955E-01  0.36294E-01
     53         16       0.90816E-03  0.64633E-03   0.00000E+00  0.00000E+00
     54         17       0.26202E-03  0.26049E-03   0.00000E+00  0.00000E+00
     55          18        0.31984E-05  0.31771E-05   0.00000E+00  0.00000E+00

        **** ENTER "1" FOR MIN. OF LFT-DAM RESIDUALS ONLY ****
        ****       USING ARB VECTOR ELEMENTS AS VARIABLES****
        **** ENTER "2" FOR LFT NULL SPACE INTRSEC. ANALY. ****
        **** ENTER "3" TO SHIFT A DESIRED SLOW EIGENVALUE ****
        **** ENTER "4" TO EXIT ELSPACE AND WRITE OPTEIG   ****
        **** ENTER "5" FOR MIN. OF LFT-DAM RESIDUALS ONLY ****
        ****             USING EIGENVALUE AS VARIABLE.    ****
        **** ENTER "6" TO WRITE SEGEIG.                   ****
        **** ENTER "7" TO EXIT ELSPACE.                   ****
        **** ENTER "0" TO EXIT ELSPACE AND WRITE XSLOW.   ****

1

        **** ENTER SLOW INDEX OF THE RESIDUAL FOR ADS ****

06

        **** ADS IS BEING ENTERED FOR ESLOW( 6) ****
        **** Z =  0.99363E+00  0.00000E+00J                ****
        **** S = -0.51123E+00  0.00000E+00J                ****
        **** ENTER SOUNDS ON THIS SLOW LFT EVECTOR  ****
        **** IN F20.13 FORMAT.                      ****
        **** UNDAMAGED EIGENVECTOR NO. 43****


        **** REAL TOLERANCE =  ****

0.001

        **** IMAG TOLERANCE =  ****

0.001

        ****       ADS COMPLETE         ****
                OBJECTIVE FUNCTION =  0.86568E+01
                2-NORM OF RESIDUAL =  0.86568E+01
                DT 2-NORM OF DIFF  =  0.38124E-02
                INFO = 0   ITER =  373
           DAM-LT OBJ =  0.00000E+00  UND-LT OBJ =  0.00000E+00
           WEIGHT FACTOR =  0.10000E-01


        **** ENTER "1" TO REENTER ADS   ****
        **** ENTER "2" TO WRITE ADSDATA ****
        **** ENTER "3" TO WRITE XSLOW   ****
        **** ENTER "4" TO WRITE SEGEIG  ****
        **** ENTER "0" TO WRITE OPTEIG  ****

1

        **** DAM-RT SLOW SUBSPACE RESIDUAL =  0.43720E-01 ****
        **** DAM-LT SLOW SUBSPACE RESIDUAL =  0.38988E+02 ****


UND EVECT  SLOW STATE   RGT-DAM RESIDUALS          LFT-DAM RESIDUALS

     38          1        0.41294E-02  0.38663E-02   0.61855E+01  0.46814E+01
     39          2        0.41349E-02  0.38703E-02   0.61855E+01  0.46814E+01
     40          3        0.31302E-02  0.28796E-02   0.86096E+01  0.54260E+01
     41          4        0.46040E-02  0.37540E-02   0.90522E-02  0.63941E-02
     42          5        0.46599E-02  0.37445E-02   0.90522E-02  0.63941E-02
     43          6        0.25482E-02  0.24184E-02   0.86568E+01  0.59235E+01
     44          7        0.27541E-02  0.26016E-02   0.68938E+01  0.48743E+01
     45          8        0.41913E-02  0.36321E-02   0.10488E+00  0.74161E-01
     46          9        0.39874E-03  0.37975E-03   0.15057E+01  0.12600E+01
     47         10        0.11465E-03  0.89779E-04   0.20671E-01  0.14616E-01
     48         11        0.15326E-02  0.13811E-02   0.11536E-01  0.81495E-02
```

```
      49          12        0.98419E-02  0.93750E-02   0.69335E+00  0.69318E+00
      50          13        0.52770E-04  0.47872E-04   0.27317E-01  0.19316E-01
      51          14        0.35123E-05  0.34142E-05   0.23157E-01  0.23054E-01
      52          15        0.42532E-03  0.42198E-03   0.51955E-01  0.36294E-01
      53          16        0.93292E-03  0.67828E-03   0.00000E+00  0.00000E+00
      54          17        0.26201E-03  0.26048E-03   0.00000E+00  0.00000E+00
      55          18        0.31981E-05  0.31768E-05   0.00000E+C0  0.00000E+00

       **** ENTER "1" FOR MIN. OF LFT-OAM RESIDUALS ONLY ****
       ****        USING ARB VECTOR ELEMENTS AS VARIABLES****
       **** ENTER "2" FOR LFT NULL SPACE INTRSEC. ANALY. ****
       **** ENTER "3" TO SHIFT A DESIREO SLOW EIGENVALUE ****
       **** ENTER "4" TO EXIT ELSPACE ANO WRITE OPTEIG   ****
       **** ENTER "5" FOR MIN. OF LFT-OAM RESIOUALS ONLY ****
       ****            USING EIGENVALUE AS VARIABLE.     ****
       **** ENTER "6" TO WRITE SEGEIG.                   ****
       **** ENTER "7" TO EXIT ELSPACE.                   ****
       **** ENTER "0" TO EXIT ELSPACE ANO WRITE XSLOW.   ****

 1

       **** ENTER SLOW INOEX OF THE RESIOUAL FOR AOS ****

 09

       **** AOS IS BEING ENTERED FOR ESLOW( 9) ****
       **** Z =  0.99978E+00  0.00000E+00J                ****
       **** S = -0.17602E-01  0.00000E+00J                ****
       **** ENTER BOUNOS CN THIS SLOW LFT EVECTCR  ****
       **** IN F20.13 FORMAT.                      ****
       **** UNDAMAGED EIGENVECTOR NO. 46****


       **** REAL TOLERANCE =  ****

 0.001

       **** IMAG TOLERANCE =  ****

 0.001

       ****       AOS COMPLETE        ****
               OBJECTIVE FUNCTION =  0.13505E+01
               2-NORM OF RESIOUAL =  0.13505E+01
               OT 2-NORM OF OIFF  =  0.28475E-02
               INFO = 0    ITER =  472
         OAM-LT OBJ =  0.00000E+00  UND-LT OBJ =  0.00000E+00
         WEIGHT FACTOR =  0.10000E-01


       **** ENTER "1" TO REENTER AOS   ****
       **** ENTER "2" TO WRITE AOSDATA ****
       **** ENTER "3" TO WRITE XSLOW   ****
       **** ENTER "4" TO WRITE SEGEIG  ****
       **** ENTER "0" TO WRITE OPTEIG  ****

 1

       **** DAM-RT SLOW SUBSPACE RESIOUAL =  0.43731E-01 ****
       **** OAM-LT SLOW SUBSPACE RESIOUAL =  0.38833E+02 ****


 UND EVECT  SLOW STATE   RGT-OAM RESIOUALS          LFT-OAM RESIOUALS

      38           1        0.41292E-02  0.38661E-02   0.61855E+01  0.46814E+01
      39           2        0.41348E-02  0.38702E-02   0.61855E+01  0.46814E+01
      40           3        0.31302E-02  0.28796E-02   0.86096E+01  0.54260E+01
      41           4        0.46040E-02  0.37540E-02   0.90522E-02  0.63941E-02
      42           5        0.46720E-02  0.37594E-02   0.90522E-02  0.63941E-02
      43          6       0.25481E-02  0.24183E-02   0.86568E+01  0.59235E+01
      44          7       0.27540E-02  0.26015E-02   0.68938E+01  0.48743E+01
      45          8       0.41914E-02  0.36321E-02   0.10488E+00  0.74161E-01
      46          9       0.39872E-03  0.37973E-03   0.13505E+01  0.12601E+01
      47         10       0.11465E-03  0.89782E-04   0.20671E-01  0.14616E-01
      48         11       0.15326E-02  0.13811E-02   0.11536E-01  0.81495E-02
      49         12       0.98413E-02  0.93745E-02   0.69335E+00  0.69318E+00
      50         13       0.52771E-04  0.47873E-04   0.27317E-01  0.19316E-01
      51         14       0.35123E-05  0.34143E-05   0.23157E-01  0.23054E-01
      52         15       0.42532E-03  0.42198E-03   0.51955E-01  0.36294E-01
      53         16       0.93300E-03  0.67827E-03   0.00000E+00  0.00000E+00
      54         17       0.26201E-03  0.26048E-03   0.00000E+00  0.00000E+00
      55         18       0.31981E-05  0.31768E-05   0.00000E+00  0.00000E+00

        **** ENTER "1" FOR MIN. OF LFT-DAM RESIDUALS ONLY ****
        ****       USING ARB VECTOR ELEMENTS AS VARIABLES****
        **** ENTER "2" FOR LFT NULL SPACE INTRSEC. ANALY. ****
        **** ENTER "3" TO SHIFT A DESIRED SLOW EIGENVALUE ****
        **** ENTER "4" TO EXIT ELSPACE AND WRITE OPTEIG   ****
        **** ENTER "5" FOR MIN. OF LFT-DAM RESIDUALS ONLY ****
        ****           USING EIGENVALUE AS VARIABLE.      ****
        **** ENTER "6" TO WRITE SEGEIG.                   ****
        **** ENTER "7" TO EXIT ELSPACE.                   ****
        **** ENTER "0" TO EXIT ELSPACE AND WRITE XSLOW.   ****

 4

        *ENTER "1" TO CALC RESIDS FOR RECONSTRUCTED SPACE*
        *ELSE "0".                                       *

 1

        **** DAM-RT SLOW SUBSPACE RESIDUAL =  0.43731E-01 ****
        **** DAM-LT SLOW SUBSPACE RESIDUAL =  0.91277E+02 ****


 UND EVECT  SLOW STATE   RGT-DAM RESIDUALS          LFT-DAM RESIDUALS
          .
      38          1       0.41292E-02  0.38661E-02   0.21257E+02  0.20396E+02
      39          2       0.41348E-02  0.38702E-02   0.22255E+02  0.21394E+02
      40          3       0.31302E-02  0.28796E-02   0.68949E+01  0.50227E+01
      41          4       0.46040E-02  0.37540E-02   0.90448E-02  0.63947E-02
      42          5       0.46720E-02  0.37594E-02   0.91533E-02  0.63948E-02
      43          6       0.25481E-02  0.24183E-02   0.16404E+02  0.15390E+02
      44          7       0.27540E-02  0.26015E-02   0.92555E+01  0.76545E+01
      45          8       0.41914E-02  0.36321E-02   0.10489E+00  0.74163E-01
      46          9       0.39872E-03  0.37973E-03   0.11592E+02  0.11384E+02
      47         10       0.11465E-03  0.89782E-04   0.20672E-01  0.14617E-01
      48         11       0.15326E-02  0.13811E-02   0.11574E+01  0.81495E-02
      49         12       0.98413E-02  0.93745E-02   0.78337E+00  0.69219E+00
      50         13       0.52771E-04  0.47873E-04   0.27317E-01  0.19315E-01
      51         14       0.35123E-05  0.34143E-05   0.26001E+01  0.25697E+01
      52         15       0.42532E-03  0.42198E-03   0.51991E-01  0.36295E-01
      53         16       0.93300E-03  0.67827E-03   0.00000E+00  0.00000E+00
      54         17       0.26201E-03  0.26046E-03   0.00000E+00  0.00000E+00
      55         18       0.31981E-05  0.31768E-05   0.00000E+00  0.00000E+00

        **** ENTER "1" FOR MIN. OF LFT-DAM RESIDUALS ONLY ****
        ****       USING ARB VECTOR ELEMENTS AS VARIABLES****
        **** ENTER "2" FOR LFT NULL SPACE INTRSEC. ANALY. ****
        **** ENTER "3" TO SHIFT A DESIRED SLOW EIGENVALUE ****
        **** ENTER "4" TO EXIT ELSPACE AND WRITE OPTEIG   ****
        **** ENTER "5" FOR MIN. OF LFT-DAM RESIDUALS ONLY ****
        ****           USING EIGENVALUE AS VARIABLE.      ****
        **** ENTER "6" TO WRITE SEGEIG.                   ****
        **** ENTER "7" TO EXIT ELSPACE.                   ****
        **** ENTER "0" TO EXIT ELSPACE AND WRITE XSLOW.   ****

 4
```

```
        *ENTER "1" TO CALC RESIDS FOR RECONSTRUCTED SPACE*
        *ELSE "0".                                       *

0
R; T=151.47/154.81 13:08:48
RECONF
VS FORTRAN COMPILER ENTERED.  13:09:04

**MAIN** END OF COMPILATION 1 ******

**CMATML** END OF COMPILATION 2 ******

**RANKD** END OF COMPILATION 3 ******

**WRMATD** END OF COMPILATION 4 ******

**SROOT** END OF COMPILATION 5 ******
VS FORTRAN COMPILER EXITED.   13:09:11


LOADING RECONF AND ORACLSA
EXECUTION BEGINS...

        ENTER "1" TO WRITE UNDAMAGED EIGENSTRUCTURE,&
                            SYST. MATRICES.(READ "RECONF" DATA A1,
                            "UNDIEG","UNDABC",& "CLM".
        ENTER "2" TO READ UNDAMAGED  EIGENSTRUCTURE AND
                            WRITE SLOW RT SUBPACE DATA FOR USE BY
                            "ERSPACE" (WRITE "ERSPACE" DATA A1)
        ENTER "3" TO COMPUTE RECONFIGURED GAINS.
                            (WRITE "FRECON" DATA A1)
        ENTER "4" TO COMPUTE RECONFIG. EIGENSTRUCTURE
                            AND TIME RESPONSE (OPTMATD,OPTPLOT)
        ENTER "5" TO COMPUTE Q AND WRITE FILE 12
                            "FXACT" DATA.

3

        **** ENTER "1" TO DISPLAY DESIRED EIGENSTRCTURE****
        **** ENTER "0" OTHERWISE.                      ****

0

        **** CALLING LSVDF FOR B ****


        **** LSVDF OF "B" COMPLETE ****

          **** RANK(B) = 10         ****

        **** CALLING LSVDF FOR C ****


        **** LSVDF OF "C" COMPLETE ****

        **** RANK(C) = 18         ****

R; T=8.34/9.72 13:09:41
RECONF
VS FORTRAN COMPILER ENTERED.  13:09:45

**MAIN** END OF COMPILATION 1 ******

**CMATML** END OF COMPILATION 2 ******

**RANKD** END OF COMPILATION 3 ******

**WRMATD** END OF COMPILATION 4 ******
```

```
**SROOT** END OF COMPILATION 5 ******
VS FORTRAN COMPILER EXITED.   13:09:53


LOADING RECONF AND ORACLSA
EXECUTION BEGINS...

        ENTER "1" TO WRITE UNDAMAGED EIGENSTRUCTURE,&
                           SYST. MATRICES.(READ "RECONF" DATA A1.
                           "UNDIEG","UNDABC",& "CLM".
        ENTER "2" TO READ UNDAMAGED  EIGENSTRUCTURE AND
                           WRITE GLOW RT SUBSPACE DATA FOR USE BY
                           "ERSPACE" (WRITE "ERSPACE" DATA A1)
        ENTER "3" TO COMPUTE RECONFIGURED GAINS.
                           (WRITE "FRECON" DATA A1)
        ENTER "4" TO COMPUTE RECONFIG. EIGENSTRUCTURE
                           AND TIME RESPONSE (OPTMATD,OPTPLOT)
        ENTER "5" TO COMPUTE Q AND WRITE FILE 12
                           "FXACT" DATA.

4

        **** ENTER "1" TO DISPLAY RECONFIGURED          ****
        **** EIGENSTRUCTURE. ELSE ENTER "0".            ****

0

        **** DAMAGED FEEDBACK GAIN MATRIX ****

          COL  1-9

0.21435E+00 -0.13143E+00  0.00000E+00  0.00000E+00  0.24000E-01  0.00000E+00 -0.25739E-02  0.74813E-03 -0.19141E-01
0.21435E+00 -0.13143E+00  0.00000E+00  0.00000E+00 -0.24000E-01  0.00000E+00 -0.25739E-02  0.74813E-03 -0.19141E-01
0.00000E+00  0.00000E+00  0.20948E-01  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
0.00000E+00  0.00000E+00  0.20948E-01  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
0.00000E+00  0.00000E+00  0.11028E-01  0.00000E+00  0.19200E-01  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
0.00000E+00  0.00000E+00  0.11028E-01  0.00000E+00 -0.19200E-01  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.60000E-01  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00 -0.60000E-01  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
0.00000E+00  0.00000E+00  0.00000E+00  0.68937E+00  0.38807E+00  0.16500E+02  0.00000E+00  0.00000E+00  0.00000E+00
0.00000E+00  0.00000E+00  0.00000E+00  0.68937E+00  0.38807E+00  0.16500E+02  0.00000E+00  0.00000E+00  0.00000E+00

        COL  10 - 18

  -0.19141E-01 -0.21079E+00 -0.38282E-01  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
0.19141E-01 -0.21079E+00 -0.38282E-01  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
0.00000E+00  0.00000E+00  0.00000E+00  0.41235E-01  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
0.00000E+00  0.00000E+00  0.00000E+00  0.41235E-01  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.21883E-01  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.21883E-01  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00 -0.85636E-02 -0.49395E-02  0.78040E-04  0.31750E-02
0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00 -0.85636E-02 -0.49395E-02  0.78040E-04  0.31750E-02

        **** FTILDA ****

          COL  1-9

    0.21414E+00 -0.13102E+00  0.44209E-03  0.86601E-06  0.24001E-01  0.38586E-05 -0.26040E-02  0.74733E-03 -0.19364E-01
0.21414E+00 -0.13102E+00  0.44171E-03 -0.32910E-06 -0.24001E-01  0.42988E-05 -0.26040E-02  0.74733E-03 -0.19364E-01
0.35389E-02  0.74240E-02  0.26697E-01 -0.90519E-06 -0.29955E-06 -0.19859E-04 -0.50029E-03 -0.13901E-04 -0.37204E-02
0.35390E-02  0.74241E-02  0.26698E-01 -0.90517E-06 -0.29954E-06 -0.19859E-04 -0.50030E-03 -0.13902E-04 -0.37205E-02
0.59863E-02 -0.93988E-02  0.47866E-02 -0.10017E-04  0.19203E-01 -0.22555E-03  0.51752E-03  0.22812E-04  0.38487E-02
0.59859E-02 -0.93978E-02  0.47956E-02 -0.14390E-05 -0.19204E-01  0.20425E-04  0.51748E-03  0.22812E-04  0.38484E-02
0.26492E-07 -0.18214E-06 -0.15221E-05 -0.16780E-05  0.60005E-01 -0.52361E-04 -0.30742E-08  0.28945E-10 -0.35545E-07
0.26453E-07  0.18196E-06  0.15203E-05  0.16764E-05 -0.60005E-01  0.52332E-04  0.30769E-08 -0.28758E-10  0.35566E-07
0.14968E-05 -0.63927E-05 -0.43987E-04  0.68943E+00  0.38810E+00  0.16500E+02  0.89218E-07  0.72124E-08  0.64980E-06
```

```
0.14751E-05 -0.63450E-05 -0.43932E-04  0.68943E+00  0.38810E+00  0.16500E+02  0.86011E-07  0.71250E-08  0.62595E-06

       COL 10 - 18

  -0.19364E-01 -0.21025E+00 -0.38728E-01  0.24654E-04  0.17951E-04  0.10460E-07  0.54048E-08 -0.22918E-08 -0.84498E-08
0.19364E-01 -0.21025E+00 -0.38728E-01  0.24659E-04  0.17918E-04  0.13479E-07  0.65791E-08 -0.22551E-08 -0.69394E-08
0.37204E-02  0.93827E-02 -0.74407E-02  0.41299E-01  0.43515E-03 -0.30406E-07 -0.16204E-07  0.42507E-08  0.15983E-07
0.37205E-02  0.93829E-02 -0.74409E-02  0.41299E-01  0.43516E-03 -0.30405E-07 -0.16203E-07  0.42510E-08  0.15985E-07
0.38487E-02 -0.12294E-01  0.76965E-02 -0.57057E-04  0.21408E-01 -0.56211E-07 -0.28421E-07 -0.29828E-08  0.23661E-07
0.38484E-02 -0.12293E-01  0.76959E-02 -0.57047E-04  0.21409E-01 -0.16108E-06 -0.10392E-06 -0.26047E-08  0.48417E-07
0.35545E-07 -0.97159E-07 -0.47360E-07 -0.19057E-08 -0.17158E-06  0.15267E-07  0.12369E-07 -0.79990E-10 -0.49395E-08
0.35566E-07  0.97003E-07  0.47387E-07  0.19020E-08  0.17133E-06 -0.15338E-07 -0.12425E-07  0.79930E-10  0.49528E-08
0.64980E-06 -0.56788E-05  0.19318E-05 -0.15884E-06 -0.59742E-05 -0.85640E-02 -0.49398E-02  0.78041E-04  0.31752E-02
0.62595E-06 -0.56191E-05  0.18841E-05 -0.15834E-06 -0.59690E-05 -0.85637E-02 -0.49398E-02  0.78045E-04  0.31753E-02

       **** ENTER "1" FOR TIME RESPONSE CALC. ****
       **** ENTER "0" OTHERWISE.              ****

0
R: T=7.57/9.04 13:10:25
```

LIST OF REFERENCES

1. Safanov, M.G., *Stability and Robustness of Multivariable Feedback Systems*, The MIT Press, 1980.

2. Ogata, K., *Modern Control Engineering*, Prentice-Hall, 1970.

3. McRuer, D., Ashkenas, I., and Graham, D., *Aircraft Dynamics and Automatic Control*, Princeton University Press, 1973.

4. Anderson, B.D.O. and Moore, J.B., *Linear Optimal Control*, Prentice - Hall, 1971.

5. Kwakernaak, H. and Sivan, R., *Linear Optimal Control Systems*, Wiley - Interscience, 1972.

6. Athans, M., "The Role and Use of the Stochastic Linear-Quadratic-Gaussian Problem in Control System Design," *IEEE Transactions on Automatic Control* Vol. AC - 10, No. 6, December 1971.

7. Owens, D.H., *Feedback and Multivariable Systems*, Peter Peregriness, LTD, 1978.

8. Gordon, V.C., *Utilization of Numerical Optimization Techniques in the Design of Multi - Input Multi - Output Control Systems*, Ph.D. Thesis, Naval Postgraduate School, Monterey, California, September 1984.

9. Chow, W.K., *Automated Pole Placement Algorithm for Multivariable Optimal Control Synthesis*, M.S. Thesis, Naval Postgraduate School, Monterey, California, September 1985.

10. Moore, B.C., "On the Flexibility Offered by State Feedback in Multivariable Systems Beyond Closed Loop Eigenvalue Assignment," *IEEE Transactions on Automatic Control*, pp. 689 - 692, October 1976.

11. Srinathkumar, S., *Spectral Characterization of Multi-Input Dynamic Systems*, Ph.D. Dissertation, Oklahoma State University, 1976.

12. Klein, G. and Moore, B.C., "Eigenvalue-Generalized Eigenvector Assignment with State Feedback," *IEEE Transactions on Automatic Control*, pp. 140 - 141, February 1977.

13. Srinathkumar, S., "Eigenvalue Eigenvector Assignment Using Output Feedback," *IEEE Transactions on Automatic Control*, Vol. AC - 23, pp. 79 - 81, February 1978.

14. Sebakhy, O.A. and Abdel - Moneium, T.M., "State Regulation in Linear Discrete - Time Systems in Minimum Time," *IEEE Transactions on Automatic Control*, Vol. AC - 24, No. 1, February 1979.

15. Klein, G., "An Eigenstructure Approach to Noninteracting Control Synthesis," *Proceedings of the 20th IEEE Conference on Decision and Control*, pp. 688 - 689, 1981.

16. Wonham, W.M., *Linear Multivariable Control: A Geometrical Approach*, Springer - Verlag, 1979.

17. Andry, A.N., Shapiro, E.Y., and Chung, J.C., "Eigenstructure Assignment for Linear Systems," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES 19, No. 5, September 1983.

18. Magni, J.F. and Herail, O., "Disturbance Decoupling by Eigenvector Assignment. An Application to Gust Alleviation," *Proceedings of the 23rd Conference on Decision and Control, Las Vegas, Nevada*, pp. 616 - 619, December 1984.

19. Eastman, W.L. and Bossi, J.A., "Design of linear quadratic regulators with assigned eigenstructure," *International Journal of Control*, Vol. 39, No. 4, pp. 731 - 742, 1984.

20. Solheim, O.A., *International Journal of Control*, Vol. 15.

21. Kautsky, J., Nichols, N.K., and Van Dooren, P., "Robust Pole Assignment in linear state feedback," *International Journal of Control*, Vol. 41, No. 5, pp. 1129 - 1155, May 1985.

22. Golub, G.H. and Van Loan, C.F., *Matrix Computations*, The John Hopkins University Press, 1983.

23. Sobel, K.M. and Shapiro, E.Y., "Eigenstructure Assignment: A Tutorial - Part I Theory," *Proceedings of the 1985 American Control Conference*, 1985.

24. Sobel, K.M. and Shapiro, E.Y., "A Tutorial - Part II Applications," *Proceedings of the 1985 American Control Conference*, 1985.

25. Mielke, R.R., Carraway, P.I., and Marefat, M., "Interactive Software for Spectral Assignment," *Proceedings of the 1985 American Control Conference*, 1985.

26. Liebst, B.S. and Garrard, W.L., "Applications of Eigenspace Techniques to Design of Aircraft Control Systems," *Proceedings of the 1985 American Control Conference*, 1985.

27. White, J.E. and Speyer, J.L., "Detection Filter Design by Eigensystem Assignment," *Proceedings of the 1985 American Control Conference*, 1985.

28. Fletcher, L.R., Kautsky, J., Kolka, G.K.G., and Nichols, N.K., "Some necessary and sufficient conditions for eigenstructure assignment," *International Journal of Control*, Vol. 42, No. 6, pp. 1457 - 1468, December 1985.

29. Sogaard-Anderson, P., Trostmann, E., and Conrad, F., "Eigenstructure and residuals in Multivariable state-feedback Design," *International Journal of Control*, Vol. 44, No.7, pp.427 - 439, August 1986.

30. Gavito, V.F. and Collins, D.J., "Applications of Eigenstructure Assignment to Design of Robust Decoupling Controllers in MIMO Systems, AIAA Paper No. 86-2246," *AIAA Guidance and Control Conference, 1986, A Collection of Technical Papers*, 1986.

31. Nashed, M. Zuhair, and others, *Generalized Inverses and Applications*, Academic Press, 1976.

32. Vanderplaats, G.H., *Numerical Optimization Techniques For Engineering Design with Applications*, Mc-Graw Hill Book Company, 1984.

33. Vanderplaats, G.H., *ADS - A Fortran Program for Automated Design Synthesis*, The Naval Postgraduate School, Monterey, California, 1984.

34. Mukhopakhyay, V. and Newsom, J., *Applications of Matrix Singular Value Margins of Multiloop Systems*, NASA TM-84524, July 1982.

35. Lehtomaki, N.L., *Practical Robustness Measures In Multivariable Control System Analysis*, PhD. Thesis, The Massachusetts Institute of Technology, 1981.

36. Sandell, N.R., and others, *Multivariable Stability Margins for Vehicle Flight Control Systems*, Alphatec, Inc. TR-121, December 1981.

37. Franklin, G.F. and Powell, J.D., *Digital Control of Dynamic Systems*, Addison-Wesley Publishing Company, 1981.

38. Doyle, J.C. and Stein, G., "Multivariable Feedback Design: Concepts for a Classical Modern Synthesis," *IEEE Transactions on Automatic Control*, Vol. AC-26, No. 1, pp. 4 - 16, February 1981.

39. Rojek, F., *Development of a Mathematical Model That Simulates the Longitudinal and Lateral-Directional Response of the F/A-18 For the Study of Flight Control Reconfiguration*, M.S. Thesis, The Naval Postgraduate School, Monterey, California, September 1986.

40. Weiss, J.L., Looze, D.P., and Eterno, J.S., "Simulation Results of Automatic Restructurable Flight Control System Concepts,AIAA Paper No. 86-2032," *AIAA Guidance and Control Conference, 1986, A Collection of Technical Papers*, pp.190 - 197, 1986.

41. McDonnell Douglas Corporation Report No. MDC A7813, *F/A-18A Flight Control System Design Report Vol. I and II, System Description and Theory of Operation*, December 1982.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Dr. D.J. Collins Office of Naval Research Branch London Box 39 FPO New York, New York 09510-0700	1
4.	Professor R.H.Franke, Code 53Fe Naval Postgraduate School Monterey, California 93943-5000	1
5.	Professor H.A. Titus, Code 62Ts Naval Postgraduate School Monterey, California 93943-5000	1
6.	Commander, Naval Air Systems Command Combat Survivabiltiy Branch Attn: Mr. Dale Atkinson (Code 5164) Washington, D.C. 20361	1
7.	Commander, Naval Air Development Center Attn: Mr. T. Jansen (Code 6013) Warminster, Pennsylvania 18974-5000	1
8.	Commander, Naval Weapons Center Attn: Mr. John Morrow (Code 338) China Lake, California 93555	1
9.	Professor Erik Trostmann Control Eingineering Institute Technical University of Denmark Building 424, DK-2800 Lyngby, Denmark	1
10.	Professor John O'Reilly Department of Electrical Engineering Industrial Control Unit University of Strathclyde Glasgow, Scotland	1
11.	CDR V.F. Gavito, Jr. 6022 Henniker Drive Houston, Texas 77041	2

NAVAL [illegible] SCHOOL
MONTEREY CALIFORNIA 93943-5002

Th
G2
c.

221014

Thesis
G25424 Gavito
c.1
Applications of eigen-
structure assignment to
design of robust mimo
decoupling controllers
and to reconfiguration
algorithm for damaged
flight control.

221014

Thesis
G25424 Gavito
c.1
Applications of eigen-
structure assignment to
design of robust mimo
decoupling controllers
and to reconfiguration
algorithm for damaged
flight control.

thesG25424
Applications of eigenstructure assignmen
3 2768 000 75892 4
DUDLEY KNOX LIBRARY